Mika Turkumäki

Tietokantamallinen laatukustannusmittari

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

20.5.2013

Metropolia

Tekijä Otsikko Sivumäärä Aika	Mika Turkumäki Tietokantamallinen laatukustannusmittari 34 sivua + 2 liitettä 20.5.2013
Tutkinto	insinööri (AMK)
Koulutusohjelma	mediatekniikka
Suuntautumisvaihtoehto	digitaalinen media
Ohjaajat	laatuvastaava Kirsi Linkola lehtori Aarne Klemetti

Insinöörityön tavoitteena oli kehittää suomalaisen hammaskuvantamislaitevalmistajan laatukustannusmittarin toimintaa ja käytännöllisyyttä luotettavan kustannusdatan tuottamiseen ja yrityksen laatuongelmien tunnistamiseen. Asiakkaan kannalta tavoitteena oli saada tuotettua nopea, varmatoiminen ja päivitettävissä oleva työkalu, joka tehostaisi laatukustannuslaskentaa ja näin edesauttaisi kustannusten pienentämistä. Projektin tavoitteet määriteltiin yhteistyössä asiakasyrityksen laatuosaston kanssa.

Työssä tutkittiin ja kehitettiin ratkaisuja, joiden avulla tietokantamallisesta laatukustannusmittarista saataisiin mahdollisimman yksinkertainen ja ymmärrettävä rakenteeltaan, jotta ohjelmointikieliä tuntemattomat voisivat itse jälkeenpäin päivittää tietokantaa. Insinöörityössä kehitettiin tietokantaan lisäksi useita automatisoituja toimintoja, jotka mahdollistavat tietokannan helpomman, varmemman ja nopeamman käytön. Tietokannan rakenne päivitettiin vastaamaan yrityksen nykyisiä tarpeita ja varmistettiin myös tietokannan helppo päivitettävyys tulevaisuuden muutoksia varten. Työssä laadittiin päivitetyt työohjeet laatukustannuslaskennan tekemiseen ja luotiin ohjeistus tietokannan käytön ongelmatilanteista, niiden ratkaisusta ja kannan päivityksestä.

Insinöörityön lopputuloksena syntyi tehokas ja helposti laajennettava työkalu yrityksen laatukustannusten mittaamiseen ja kustannusten aiheuttajien paikallistamiseen. Aikaisempaan mittariversioon verrattuna uuden version käyttö osoittautui ajallisesti yli kahdeksan kertaa nopeammaksi ja näin ollen huomattavasi edullisemmaksi. Analyysit mittarin tuottamista tuloksista saadaan tehtyä nopeasti ja automatisoidusti muodossa, joka on helppo esittää ymmärrettävästi yrityksen johdolle.

Avainsanat	laatu, laatukustannus, laaduttomuus, tietokanta, SQL, Visual Basic

Author Title Number of Pages Date	Mika Turkumäki Quality cost database development 34 pages + 2 appendices 20 May 2013
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructors	Quality Manager Kirsi Linkola Lecturer Aarne Klemetti

The goal of the thesis was to develop a poor quality cost meter for a dental imaging equipment manufacturer. The development was aimed to increase the functionality, reliability, speed and usability and to create quality cost data for quality issue identification. The goal from the client's perspective was to be able to create a fast and reliable tool that could be updated from time to time by the quality personnel. This would increase the effectiveness of quality cost identification and calculation and thus support reducing the costs. All of the goals and requests were defined and collected in cooperation with the company's quality department.

During the project, solutions were researched and developed, to make the database-based quality cost meter as simple as possible in use and in structure so that personnel without knowledge on SQL and Visual Basic programming languages could still update the database themselves, if required. During the project, a few automated procedures were developed to the database. Those procedures made it possible for the database to be used faster, easier and more reliably. The structure of the database was updated to correspond the current needs and products of the company and ensured the easy updateability also for future changes. During the project, also updated work instructions were created for quality cost calculation using the database. Also instructions for database update and error handling were created for quality personnel to ensure easy use of the database.

The result of the project was an effective and easily upgradable tool for quality cost measurement and identifying the cause for these costs. In comparison to the previous quality cost meter, the usage of the new version is over eight times faster and more reliable. This will keep the quality inspection costs down and more resources can be applied to the problem cause identification. All of the required analyzes from the data produced by the database can be made quickly and automatically in such a form which is easy to report and understand for the head of the company.

Keywords	Quality, poor quality, quality cost, database, SQL, Visual Basic

Sisällys

1 Johdanto

Insinöörityön tavoitteena on kehittää PaloDEx Group Oy:n laatukustannusmittaria, jotta yrityksen kuukausittaiset laaduttoman toiminnan aiheuttamat kustannukset voidaan helposti määritellä ja analysoida. Yrityksessä jo käytössä olevaa mittaria parannetaan vastaamaan yrityksen nykyisiä tarpeita, ja pyritään saamaan mittarin toiminta yksinkertaistettua ja osittain automatisoitua, jotta yrityksen laatuhenkilöstö kykenee siihen itse myöhemmin tekemään lisäyksiä ja ylimääräisen työn tekeminen vähenisi. Tämä kehitys olisi suuri edistysaskel, sillä yritys kuuluu kansainväliseen organisaatioon, jonka tuotteet ja toimintatavat voivat muuttua nopeasti, ja niihin tulee pystyä reagoimaan myös mittauksen puolella.

Kehityshankkeessa keskitytään ainoastaan tietokantamallisen kustannusmittarin parantamiseen ja tietokannan toiminnan automatisointiin. Prosessin lähtökohtana toimiva mitattavien kustannusten rajaus ja seuranta on yrityksessä ennalta määrätty, ja täten niiden kehitykseen tai tuotantotapoihin ei tarvitse ottaa kantaa. Näiden kustannusten mittauksen jälkeen prosessissa edetään kustannusten analysointiin, jonka toteuttaa laatuhenkilöstö ja joka myös näin ollen rajautuu ulos kehityshankkeen piiristä. Tiedon käsittely analysointia varten on siis kehityshankkeen suuri ja haastava, mutta ainoa, tutkimuskohde. Tällä voidaan kehityshanke pitää riittävän laajana, mutta samalla tarkoituksenmukaisen pituisena.

PaloDEx Group Oy:ssä, kuten kaikissa kehittyvissä ja tuottavissa yrityksissä, laatu merkitsee elinehtoa. Laatu edesauttaa yrityksen imagoa ja visiota olla luotettu ja paras vaihtoehto omalla alallaan. Jos laatu vastaa yrityksen tuotteiden lupauksia ja hinnoittelua, yritys saa myönteistä näkyvyyttä ja ylläpitää tai jopa kasvattaa asiakaskuntaansa. Laadukkaan tavaran tuottaminen ei kuitenkaan tapahdu itsestään, vaan se vaatii koko yrityksen panostusta tuottaa entistä parempaa ja kestävämpää tuotetta, suunnitteluasteelta tuotannon viimeistelyyn asti. Hyvän laadun tuottaminen on yritykselle kuitenkin usein hyvin kallista. Laatu kulkee käsi kädessä laaduttomuuden kanssa, ja jotta yrityksen tuotteiden laatua voidaan parantaa, on laaduttomuuden osuutta pienennettävä. Laaduttomuus onkin yksi laadukkaan tuotteen tuottamisen suurimmista kustannuskohteista. Jotta näitä kustannuksia voitaisiin pienentää, tulee niitä säännöllisesti mitata, laskea ja analysoida.

2 Laaduttomuus ja kustannukset yrityksen kannalta

2.1 Kohdeyritys

PaloDEx Group Oy on amerikkalaisomisteinen hampaistokuvantamiseen keskittynyt Suomessa toimiva laitevalmistaja. Yhtiö suunnittelee, valmistaa ja markkinoi tuotteitaan jopa yli 50 maahan. Tuotteiden paikallisen markkinoinnin tekevät ulkomaalaiset jakelijat, mutta kaikki muu toiminta on keskittynyt PaloDEx Groupin Tuusulan yksikköön. Yhtiö valmistaa hammaslääkärikäyttöön tarkoitettuja röntgenlaitteita, kuvalevylukijoita ja sensoreita. Yrityksen toiminnan mahdollistaa tiivis työyhteisö, joka nykyään käsittää noin 400 työntekijää. PaloDEx Group on vuosien myötä kasvanut yhdeksi maailman luotetuimmaksi ja tunnetuimmaksi hampaistokuvantamislaitteiden valmistajaksi. Yhtiöllä on jo vankka asiakaskunta, mutta kehittämällä lisää tuotteita se varmistaa jatkuvan liikevaihdon kasvun. [1.]

Yrityksen toiminnan perusta on tuottaa asiakkaille laadukkaita, huippuluokan kuvantamislaitteita, joissa teknologia kohtaa käytettävyyden. Yrityksen tarkoitus on tuottaa laadukkaita laitteita hammaslääkäreille, ottaen samalla huomioon potilaan tarpeet. PaloDEx Group pyrkii saavuttamaan korkeimman aseman alallaan ja olemaan luotettavin vaihtoehto hammasalan ammattilaisille. Tämä tarkoittaa, että tuotettujen laitteiden tulee olla niin teknisesti kuin ulkomuodollisesti erittäin laadukkaita, helppokäyttöisiä ja kestäviä. [1.]

Kaikkien yhtiön tavoittelemien ominaisuuksien yhdistävä tekijä on laatu. Laatu ja laadun seuranta ovat yrityksessä hyvin suuressa roolissa jokapäiväisessä toiminnassa. Yksikään laite ei poistu tuotantolinjalta, ennen kuin se on läpäissyt alati tarkentuvat testit, jolla varmistetaan niin tekniikan täydellinen toiminta, kuin ulkonäön virheettömyys. Koska kyseessä ovat laitteet, joilla tutkitaan potilaita ja joiden tutkimusten perusteella tehdään diagnooseja, on laitteiden oltava erittäin varmatoimisia ja luotettavia. Lääketieteen alalla toimittaessa vaatimukset ovat jo valmiiksi korkeita, mutta halutun laadun takaamiseksi yhtiö toimii vielä niitäkin tarkemmin.

Kuitenkin, vaikka laatua seurataan ja valvotaan jatkuvasti, ei ole taattua, että järjestelmä ei joskus pettäisi. Näissä tilanteissa tuotetta voidaan kuvata sanalla laaduton. Laaduttomuus on yksi suuri osatekijä minkä tahansa yrityksen kuukausittaisesta me-

noerästä. Koska PaloDEx Groupin tavoitteena on olla alansa luotetuin valmistaja, on laaduttomuudella ja siitä aiheutuvilla kustannuksilla yritykselle erittäin suurta painoarvoa. Yhtiön laatuorganisaatio laatii kuukausittain raportteja siitä, kuinka paljon kustannuksia yrityksen laaduttomuus aiheuttaa ja miltä osa-alueilta nämä kustannukset tulevat. Koska kyseessä on yritykselle arvokas asia, sille on luotu oma työnkulkunsa ja työkalut, joilla laatukustannuksia voidaan seurata.

2.2 Laaduttomuus ja sen aiheuttamat kulut

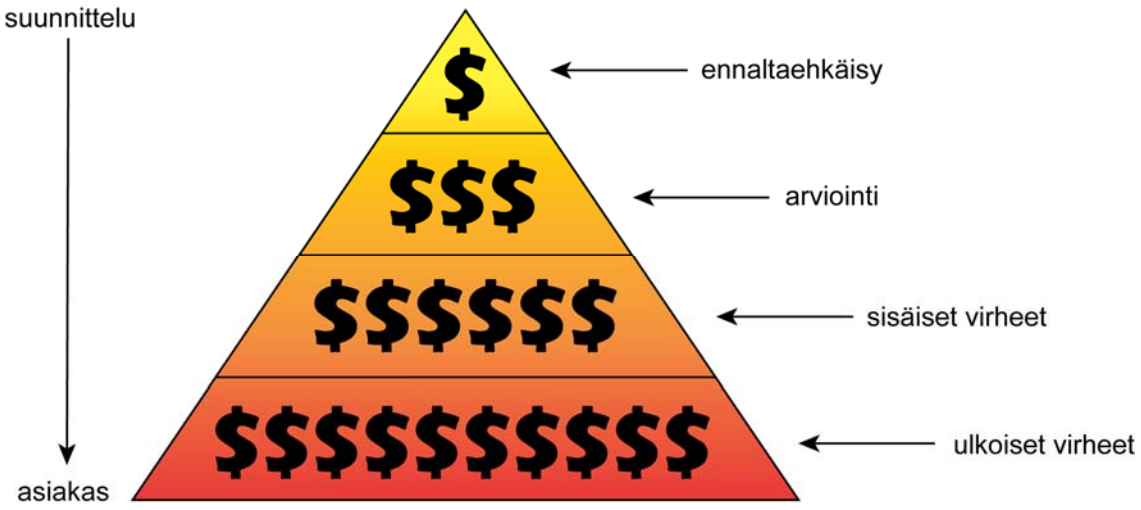
Laatu luo sanana myönteisen ja miellyttävän mielikuvan tuotteesta tai palvelusta puhuttaessa. Jotta tuote tai palvelu voisi menestyä markkinoilla, sen on oltava laadukas ja näin houkuteltava asiakkaita. Jos taas tuote on laaduton, se ei täytä sille luvattuja ja asiakkaiden vaatimia ominaisuuksia, jolloin sen myynti hiljalleen lakkaa pääsemättä kunnolla käyntiin. Tuotteen laaduttomuus vaikuttaa sen itsensä lisäksi koko yrityksen imagoon. Jos asiakkaiden luottamus yritykseen katoaa sen tuottamien laaduttomien tuotteiden myötä, ei yrityksellä ole enää markkina-arvoa joka hyvin todennäköisesti johtaa sen kaatumiseen. [2.] Laadukkaan tuotteen tuottaminen on siis yrityksen kannalta elintärkeää, mutta sen toteuttaminen vaatii useiden osastojen laadukasta ja motivoitunutta toimintaa. Laatu ei tästä syystä tule yrityksille ilmaiseksi, vaan sen eteen täytyy tehdä fyysistä työtä ja kattavaa analysointia. Laadun ja laaduttomuuden analysointi on eräs tärkeimpiä ja haastavin osa-alue laadukkaan tuotteen tuottamisessa.

Laaduttomuus on käsitteenä yksiselitteinen. Se kuvastaa tapahtumaa tai prosessia, joka ei vastaa sille asetettuja tavoitteita, olivat ne sitten fyysisiä tai toiminnallisia. Laaduttomuudeksi yleisesti käsitetään se, että tuote rikkoutuu ennen luvatun käyttöikänsä loppumista tai että tuote ei toiminnoiltaan tai ulkomuodoltaan vastaa annettua kuvaa. Tämänkaltaiset tapaukset ovat kaikille tuttuja jokapäiväisestä elämästä, on sitten kyse älypuhelimesta, jonka näyttö ei toimi, tai vajaasta maitotölkistä. Nämä ovat laaduttomuuteen viittaavia tapauksia, jotka ovat loppukäyttäjälle näkyviä ongelmia. Laaduttomuutta esiintyy jo kuitenkin paljon ennen kuin vasta valmiin tuotteen parissa.

Laatukustannusten seuranta on mittari, joka tarkastelee tuotteen tai palvelun laadun saavutuksia tai epäonnistumisia. Näihin kuuluvat kaikki vaatimukset, jotka tuotteen tai palvelun valmistaja, käyttäjät ja markkinoijat ovat asettaneet. Tarkemmin esitettynä laatukustannukset ovat kokonaissumma, joka koostuu laaduttomuuden ennaltaeh-

käisyn investoinneista, tuotteen vaateisiin mukautumisen arvioinnista sekä epäonnistumisesta täyttämään nämä vaateet. Laatukustannuksilla voidaan kuvata tuotteen oikean hinnan eroa siihen, mitä hinta olisi ilman mahdollisuutta tuotteen rikkoutumiseen tai virheisiin valmistusvaiheessa. Yrityksen sisäiset laatukustannukset voidaan jakaa neljään alaluokkaan, jotka yhdessä muodostavat yrityksen kokonaislaatukustannusrakenteen: ennaltaehkäisyn kustannukset, arvioinnin kustannukset, sisäisten virheiden aiheuttamat kustannukset ja ulkoisten virheiden aiheuttamat kustannukset. [3, s. 2–4.]

Mainitut kustannusluokat eivät ole yrityksen talouden kannalta yhteneviä, vaan yleisenä sääntönä voidaan pitää ajatusta, että laatukustannukset kasvavat aina, mitä lähemmäs asiakasta tuote siirtyy: suunnitteluasteelta tuotantoon ja jakelusta asiakkaan käyttöön. Jokaisen alaluokan yli siirtyvän laatuongelman kustannukset voivat kymmenkertaistua. Vaikka kaikki ongelmat eivät viimeiselle tasolle, eli ulkoisiin virheisiin, ylläkään, laatukustannukset kasvavat poikkeuksetta aina ongelman edetessä lähemmäs asiakasta. Laatukustannusten suuruuden suhteellinen kasvu on kuvattuna kuvassa 1, josta nähdään kustannusten moninkertaistuminen ongelman saavuttaessa asiakkaan.



Kuva 1. Laatukustannusten alaluokkien suhteelliset suuruudet (vrt. 4, s. 6).

Ennaltaehkäisyn kustannuksilla tarkoitetaan kustannuksia, jotka koostuvat toiminnoista, jotka on suunniteltu erityisesti ehkäisemään huonon laadun tuottamista. Näihin kustannuksiin voidaan sisällyttää uuden valmistettavan tuotteen laatusuunnitelman luonti, toimittajan toimituskapasiteetin arviointi, laadunparantamisprojektit ja henkilökunnan laatukoulutus. Ennaltaehkäisevät laatukustannukset eivät siis juurikaan ole yhteydessä fyysiseen tuotteeseen, vaan koostuvat niistä osa-alueista, joiden tulisi taata tuotteen

hyvä laatu valmistusprosessin aikana. Arvioinnin kustannukset koostuvat niistä mittauksista, arvioista ja auditoinneista, joilla varmistetaan, että tuote vastaa sille asetettuja vaatimuksia. Näihin kustannuksiin sisältyvät esimerkiksi saapuvan tavaran tarkastus, tuotteen lopputarkistus, testauslaitteiston kalibrointi ja näihin liittyvät materiaalikustannukset. Vaikka arvioinnin kustannukset eivät ole välttämättä fyysisiä kustannuskohteita, tulee pitää mielessä, että työ ei ole ilmaista. Työmäärä onkin siis hyvin tärkeä laatukustannuksia laskettaessa. Sisäisten ja ulkoisten virheiden aiheuttamat kustannukset ovat lähes toisiaan vastaavia. Nämä kustannukset johtuvat huonosti tuotetuista tuotteista, jotka eivät lunasta niihin asetettuja lupauksia ja vaatimuksia. Sisäiset kustannukset koostuvat yrityksen sisällä tapahtuvista laaduttomuushavainnoista ja niiden toimenpiteistä, kuten romutus, uusintatyö, uusintatarkastus ja materiaalin tarkastus. Ulkoiset kustannukset taas tulevat asiakkaan puolelta ja sisältävät asiakasvalitukset, asiakaspalautukset ja takuutoimitukset. [3, s. 5.]

Kuten voidaan havaita, laatukustannukset koostuvat suuresta määrästä erilaisia työprosesseja ja tapahtumia. Niiden kaikkien jatkuva seuraaminen ja parantaminen voi osoittautua yritykselle suureksi urakaksi ja taloudelliseksi haasteeksi. Siksi on suositeltavaa, että aloitettaessa laatukustannusten seuraamista ja mittaamista käytetään Pareto-metodia taloudellisesti suurimpien ongelmakohtien selvittämiseksi ja lähdetään ensimmäiseksi parantamaan näitä alueita laatukustannusmittausten avulla. Kun kyseiset alueet on saatu laaduttomuuden osalta korjattua, siirrytään kehittämään seuraavia, vähemmän akuutteja ongelmakohtia. [4, s. 5.] Mainittu Pareto-metodi on, laatujohtamisen tärkeimmän kehittäjän, Joseph Juranin, kehittämä ”sääntö”, joka perustuu lukuun 80/20. Käytännössä tämä voidaan käsittää laatukustannusten kannalta siten, että 20 prosenttia tuotevioista aiheuttaa 80 prosenttia ongelmallisista tuotteista. Tätä metodia voidaan soveltaa hyvin moniin aiheisiin. Esimerkiksi edellä mainitussa ongelmakohtien selvityksessä voidaan ajatella, että 80 prosenttia kustannuksista aiheutuu 20 prosentissa kokonaistyövaiheista. [5.]

Suurin osa tuotteen laatukustannuksista syntyy sille suoritettavien prosessien kautta, sen koko valmistuksen aikana. Näitä prosesseja parantamalla ja niiden suorittajien pätevöittämisellä voidaan suoraan vaikuttaa tuotteen laatuun ja täten parantaa asiakastyytyväisyyttä. Samalla, kun prosessien laatu ja toiminta paranee, pienenevät myös niistä johtuvat laatukustannukset. Tästä syystä laatukustannusmittarin avulla pyritään havaitsemaan kustannukset juuri näistä soveltuvista prosesseista. Tämän avulla voi-

daan paikallistaa ongelmakohteet ja sitä kautta poistaa ongelmien perimmäinen syy. [6, s. 66.]

Laadun ja laaduttomuuden seuranta on usein tärkeä keino pyrittäessä parantamaan yrityksen kannattavuutta, tuottavuutta ja kasvua. Laatukustannuksia pidetään tästä syystä yhtenä nykyaikaisen laatujohtamisen ja yrityksen toiminnan strategisen suunnittelun tärkeimmistä työkaluista. [7.] Kaikille laatukustannuksille on kuitenkin hyvin vaikea asettaa tarkkoja arvoja ja rajoja. Ongelmana laatukustannusten mittaamisessa pidetään sitä, että laatukustannusten seurannalla ei voida todentaa kuin osa todellisista kustannuksista. Tätä kutsutaan jäävuori-ilmiöksi, jolla tarkoitetaan sitä, että mitattavat ja mitattavissa olevat kustannukset muodostavat laatukustannusten ”jäävuoren huipun”. Suurin osa kustannuksista, joita on vaikea arvottaa, piilevät pinnan alla ja ovat usein syypäät yrityksen tai budjetin ”uppoamiseen”. [3, s. 7.] Kuvassa 2 on laatukustannusten jäävuorimalli ja alueet, jotka on määritelty yleisesti mitattaviksi ja piilokustannuksiksi.

Kuva 2. Jäävuorimallin mukainen esitys laatukustannusten aiheuttajista (vrt. 7).

Koska piilokustannuksia on vaikea ja osin lähes mahdotonta arvottaa ja laskea helpoin keinoin, ne jätetään yleensä tiedostaen pois laatukustannusmittauksen piiristä. Analysointia varten piilokustannuksille pitäisi arvioida painoarvo, mutta useimmissa tapauksissa se ei olisi lopulta yrityksen kannalta hyödyllistä, sillä suuri osa piilokustannuksista

ei ole konkreettisia. Täten niille on lähes mahdotonta analysoinnin avulla löytää korjaavia toimenpiteitä. Siitä syystä yrityksen tulisi keskittyä jäävuoren pintapuolisiin, helposti käsitettäviin kustannuslajeihin. Kun niihin keskitytään, voidaan taata paremmat tulokset helpommin ja ylläpitää henkilöstön motivaatiota laadun parantamisen kannalta. Vaikka piilokustannuksia ei sisällytetäkään tavalliseen laatukustannusmittaukseen, ne tulee tiedostaa ja ymmärtää niiden aiheuttamien menetysten mahdollisuus. Tiedostamalla piilokustannusten osuus yrityksen laatukustannusrakenteessa voidaan keskittyä paremmin prosessivirheisiin, joita parantamalla ehkäistään piilokustannusten aiheuttamia menetyksiä. [2, s. 40–42.]

2.3 Laatukustannusten laskenta ja käytännön toteutus

Luvussa 2.2 selvitetyt laatuun, laaduttomuuteen ja laatukustannuksiin liittyvät asiat tulee ottaa huomioon ja niiden tulee olla hyvin tiedostettuina, kun yritykseen aletaan suunnitella ja rakentaa laatukustannusohjelmaa. Tavallisesti tämän tekee asioista perillä oleva laatupäällikkö. Jotta laatukustannusohjelma voidaan toteuttaa, sen ohjaajalla on oltava tietoa laatukustannusmenetelmistä, ymmärrystä ohjelman hyödystä ja arvosta yritykselle sekä tahtoa selvittää yrityksen toimintaan liittyviä haasteita. [6, s. 45.] Laatukustannusten kartoittaminen vaatii datan analysointia laskentamallien avulla ja niiden soveltamista laatukustannusohjelmaan siten, mikä katsotaan yrityksen kannalta kannattavimmaksi. Perinteisinä laadun mittareina pidetään virheprosentteja ja virheiden esiintymistiheyttä. Nämä mittarit tuottavat informatiivista ja hyödyllistä tietoa asioiden parissa työskenteleville, mutta yritysmaailmassa ylintä johtoa kiinnostaa enemmän raha kuin prosenttitaulukot. Rahan liittäminen laadun mittaamiseen yhdistää organisaation ja toimii selkeänä laadun mittarina niin toiminta- kuin päätäntäelimille. [7, s. 52.] Konkretisoimalla laatukustannukset voidaan helpommin puhutella yritysjohtoa ja samalla motivoida koko yritystä kehittämään toimenpiteitä laadun parantamiseen.

Laskettaessa laatukustannuksia itse laskenta ei ole päätavoitteena. Laatukustannuksia mitattaessa pyritään aina saamaan selville yrityksen tärkeimmät kehityskohteet, jotka kuluttavat turhaan yhtiön resursseja toiminnalla, joka voisi olla korjattavissa. Laskentamenetelmällä, tai laskennan tuottamalla datalla, ei siis ole juurikaan merkitystä motivoinnin lisäksi, sillä tavoitteena on vain arvottaa laadunparantamisen kohteet tärkeysjärjestykseen. [2, s. 20.] Tärkeysjärjestyksessä yleisesti ottaen korkeimpana pidetään niitä kohteita, jotka aiheuttavat yritykselle suurimmat laatukustannuskulut. Kuvan 1 mu-

kaisesti kustannukset kasvavat sitä mukaa, mitä lähemmäs tuote siirtyy loppukäyttäjää. Tällöin laatukustannuslaskennalla voidaan paikallistaa ulkoisten virheiden perimmäiset syyt, korjata ne ja liikkua lähemmäs tuotesuunnittelua. Kun prosessit tehdään heti oikein, eivät ongelmatapaukset pääse siirtymään asiakkaalle, ja näin laatukustannukset pysyvät hallinnassa. Laatukustannusten laskennalla voidaan siis seurata laatukehitystä ja pyrkiä siirtymään virheiden korjaamisesta ennaltaehkäisevään laatutoimintaan.

Jotta yrityksen sisällä voitaisiin tehdä tarkkaa ja tehokasta laatukustannuslaskentaa, tarvitaan sitä varten oma luotettava laatukustannusten laskentajärjestelmä. Haasteena pidetään sitä, että yrityksen kustannuslaskentajärjestelmä ei välttämättä suoraan tue laatukustannusten laskentaa ja ylläpitoa, jolloin erillisen tiedon kerääminen, analysointi ja raportointi on hidasta ja vaatii paljon yrityksen resursseja. [8, s. 212–213.] Lisäksi, kuten aiemmin mainittu, kaikki laatukustannukset eivät perustu spesifisiin arvoihin, vaan niille joudutaan arvioimaan kustannukset ja määrät, esimerkiksi puhuttaessa työtunneista. Tämä vaikuttaa suoraan laatukustannusmittarissa siihen, kuinka luotettavaksi henkilöstö mittarin kokee. [2, s. 35.] Taulukossa 1 on koottuna suurimmat ongelma-alueet laatukustannusten laskennan kannalta. Kun yrityksen laatuhenkilöstö on tietoinen kustannuslaskentajärjestelmän rakentamiseen liittyvistä ongelmista ja niiden ratkaisujen suunnitteluun käytetään resursseja, on käytännön toteutuksen toimivuus varmempaa kuin ilman taustatyötä prosessia käynnistettäessä.

Taulukko 1. Laatukustannuslaskennan ongelma-alueet ja niiden selvitykset (vrt. 7, s. 64).

ONGELMA	SELITE
Tutkittavan alueen laajuuden määrittely	Mitkä yrityksen osa-alueet ja prosessit otetaan mukaan kustannuslaskennan piiriin. Lähtökohtaisesti laajuuteen ei ole yksiselitteistä vastausta.
Mittaamisen tavoitteet	Laskennan tulosten käyttötarkoitusten tulee olla ennalta suunnitellut, jotta projektilla saavutettavat tulokset eivät jää vaatimattomiksi.
Määritykset	Ei ole yksiselitteistä vastausta siihen, mitkä toiminnot ovat laatuun ja laaduttomuuteen liittyviä.
Vastuualueet	Projektille tulee olla määritelty selkeät vastuuhenkilöt, jotta sillä on mahdollisuus onnistua.
Mittauskeinot	Yrityksessä jo valmiina olevat laskentajärjestelmät eivät tue laatukustannuslaskennan tarpeita.
Johtoportaan panos	Ilman johdon tukea projektille laatukustannushankkeet yleensä epäonnistuvat.
Henkilöstön panos	Työntekijöiden tulee nähdä laskennasta ja laadun parantamisesta aiheutuvia hyötyjä, muuten heillä ei ole motivaatiota tehdä työtä sen eteen.

Tarkkuus	Usein yrityksen laatukustannusten mittauksen kyvykkyys on niin heikko, että tulokset eivät ole täysin luotettavia.
Toteutus	Laatukustannuslaskentaprojektin aloitus on usein vaativaa, ja jatkuvasti kehittyvän prosessin kautta luotettavien tulosten saaminen voi kestää vuosia.
Vertailu	Laatukustannuksia ei voida suoraan vertailla eri yritysten, eikä aina edes eri aikakausien välillä.

Aloitettaessa laatukustannuslaskentaprojektia tulee määritellä projektin tavoite ja tarkoitus laadun parantamisen kannalta. Kun laatukustannuksia lasketaan ja analysoidaan, tulee muistaa, että koko mittarin toiminta perustuu siihen, ettei pyritä vain löytämään virheitä ja korjaamaan niitä. Tarkoituksena on pyrkiä etsimään perimmäiset syyt näille virheille ja estämään niiden uudelleentapahtumisen mahdollisuus. Käytännön toteutus aloitetaan suunnitelman ja tavoitteiden ollessa selvillä. Tällöin analysoitavaa ja laskettavaa dataa tulee alkaa kerätä lähteistä, jotka vastaavat laatukustannusohjelmalle asetettuja rajoja ja vaatimuksia. Toimivan laatukustannusten laskentaohjelman perusperiaatteena on, että laatutietojen raportoijat tulevat tärkeysjärjestyksessä ensimmäisenä. Tämä tarkoittaa, että laatukustannuksiin liittyvän datan kerääminen ja hankinta on tehtävä asianomaiselle henkilölle mahdollisimman helpoksi. Laatukustannuksia mitattaessa tietovastaava hakee ja syöttää laskujärjestelmään edellisen laskentakauden laatukustannusdatan. Tieto, jonka tietovastaava hankkii, voi sisältää paljon eri yksiköitä, jotka lopulta laskentaohjelman avulla muunnetaan johtoporrasta miellyttäväksi konkreettiseksi arvoksi, eli rahaksi. [9.]

Tämän laskennan lähtökohtana on lisäarvoa tuottamattomien resurssien, kuten materiaalien ja työmäärän, arvottaminen. Kun näille resursseille on määrätty kustannukset ja tiedetään ongelmien esiintymistiheys raportoinnin perusteella, voi ongelmien aiheuttamat kustannukset laskea yksinkertaisella yhtälöllä: resurssin yksikköhinta * esiintymistiheys. Laskentaohjelman avulla luotua dataa tulee koskea myös sääntö, että siitä muodostuvan raportointimateriaalin on oltava sellaisessa muodossa, jota voidaan helposti seurata ja ymmärtää. Raporttien käyttäjien tulee siis ymmärtää näkemänsä saadakseen tietoa siitä, mitä he voivat jälkikäteen käyttää hyödykseen laadun parantamisessa. [9.] Esimerkiksi jos tiedetään, että tuotteen romutus vie yhden työtunnin, jonka perushinta on 45 € ja uusi materiaali osaan maksaa 300 € ja näitä romutuksia raportoinnin mukaan tapahtuu kolme kertaa viikossa, tästä lähtökohdasta voidaan laskentaohjelmalla helposti luoda kustannus kuukausittaiselle romutukselle: (45 € + 300 €) * 3 * 4, josta saadaan 4 140 €. Tämä luku raportoituna on helposti kaikkien ymmärrettävissä ja vertailtavissa edellisten kuukausien tuloksiin.

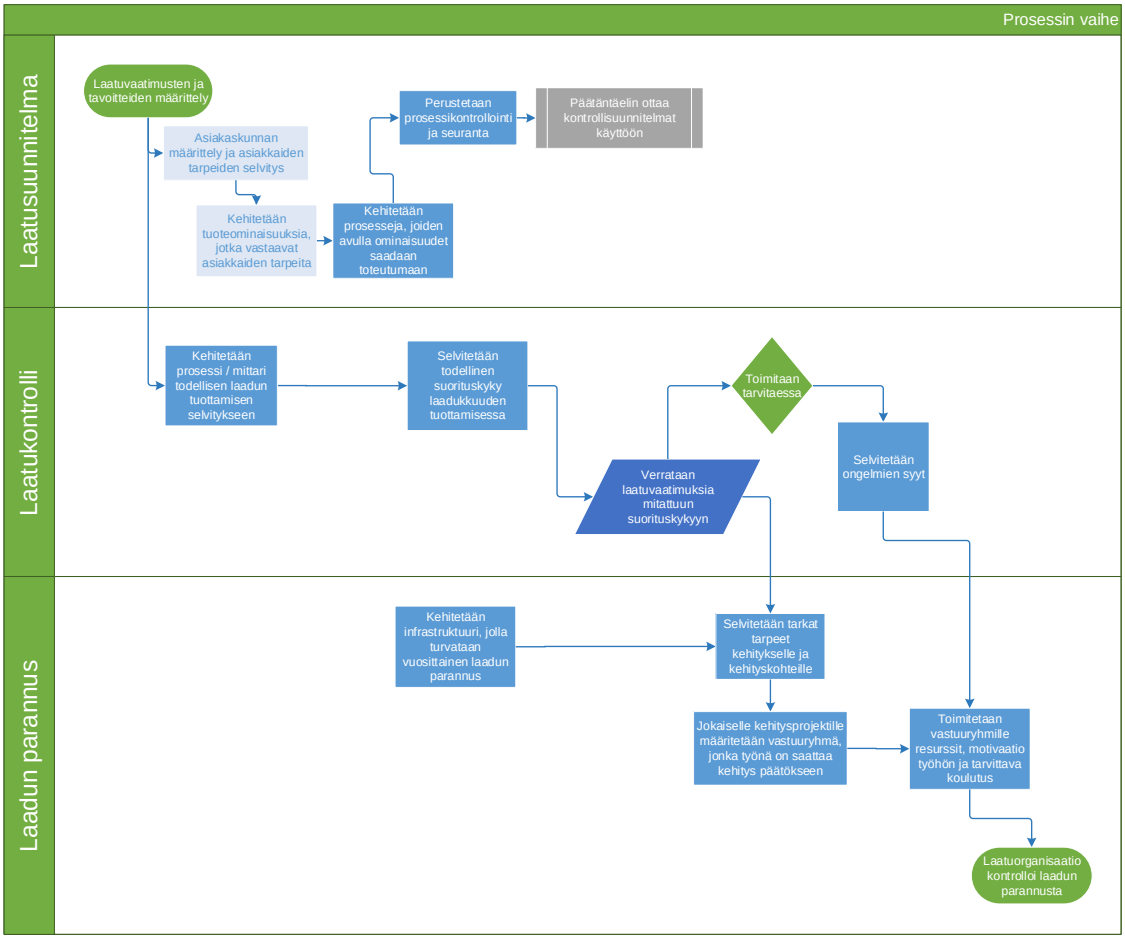
Näitä konkreettisia arvoja sisältäviä raportteja laaditaan ja analysoidaan ja päätetään toimenpiteitä niiden perusteella kuukausittain. Näin saadaan suhteutettua tasaisella intervallilla laatukustannuksen kehittymistä kuluvan vuoden budjettiin, ja sen perusteella voidaan havaita, onko laaduttomuuden parantamisessa edetty laatukustannusohjelman suunnitelman mukaisesti. Laatukustannuksia analysoitaessa tulee verrata lasketun kuukauden laatukustannuksia sille budjetoituihin kustannuksiin, jos yritys on asettanut kuukausittaiset laatukustannustavoitteet. Mikäli kustannukset ylittävät budjetoidun osuuden, tulee poikkeamien syyt analysoida tarkemmin ja pyrkiä löytämään niihin parannuskeino. [9.]

Parannuskeinoa etsittäessä törmätään väistämättä rahaongelmaan. Johdon tulee päättää toimenpiteistä ja niiden tarpeellisuudesta. Jos ongelma aiheuttaa yritykselle kustannuksia 200 € kuukaudessa ja sen korjaaminen maksaisi 5 000 €, onko sen korjaamiseen varaa ja onko sen tuoma voitto pitkällä aikavälillä niin suuri, että korjaustoimenpiteitä kannattaa lähteä toteuttamaan? Tässä tapauksessa tulee muistaa, että korjaava toimenpide tulee yritykselle kalliiksi vain kerran, kun taas korjaavan toimenpiteen tekemättä jättäminen maksaa yhä uudelleen. [3, s. 72.] Jotta toimenpiteitä voitaisiin suorittaa, tulee raportin analyysin perusteella etsiä ongelman perimmäinen syy. Tähän voidaan soveltaa useita erilaisia tutkimusmenetelmiä, kuten yleisesti käytössä olevaa, alun perin 1970-luvulla Toyotan kehittämää viiden miksi-kysymyksen menetelmää.

Perimmäisen syyn etsiminen viiden miksi-kysymyksen avulla perustuu siihen, että ongelmasta voidaan kysymysten ja niiden vastausten avulla kuoria kerroksia ja lopulta paljastaa perimmäinen syy. Käytännössä menetelmä toimii siten, että ongelmaa pohdittaessa kysytään syytä ongelmaan: ”Miksi tämä tapahtui?” tai ”Mikä aiheutti tämän ongelman?” Kun kysymykseen on löydetty vastaus, jatketaan kysymällä kysymys uudelleen, kohdistettuna edellisen kysymyksen vastaukseen. Tätä sykliä jatketaan, kunnes ongelman todellinen perimmäinen syy on löydetty. Menetelmän avulla voidaan nopeasti selvittää ongelman perimmäinen syy, jotta resursseja osataan ohjata juuri sille prosessin osa-alueelle, joka kysymysten avulla on saatu selvitettyä. [10.]

Jotta laatukustannusten laskennalle olisi yritykselle käyttöä, tulee laskentadata analysoida ja analyysin perusteella tehdä muutoksia toimintatapoihin ja organisaatioon. Joseph Juran on esittänyt niin kutsutun Juranin trilogian, jonka avulla voidaan nähdä koko prosessi, joka yrityksen tulee huomioida ja toteuttaa, jotta laadun parantaminen olisi mahdollista. [11.] Kuvassa 3 on mallinnettu uimaratamallilla Juranin trilogian mukainen

prosessivaiheistus, jolla voidaan taata laaduttomuuden väheneminen ja jatkuva laadun parantuminen.



Kuva 3. Juranin trilogia laaduttomuuden estämiseksi prosessikaaviona (vrt. 11).

Kuten kuvasta 3 nähdään, yrityksen tulee alkaa laadun parantaminen määrittelemällä omat vaatimuksensa hyvälle laadulle. Tämän jälkeen vaatimuksia tulee verrata laatukustannusmittauksella saatuihin todellisiin tuloksiin ja organisoida toimenpiteitä yrityksessä näiden tulosten mukaisesti. Lopulta, jotta pysyvää laadunparannusta voidaan edistää, laatuorganisaation tulee valvoa ja kontrolloida jatkuvasti, että tehtyjä muutoksia ja parannuksia noudatetaan ja että työntekijät on koulutettu tekemään näitä töitä. Kun yritys tukee ja noudattaa kaikkia kolmea laatuprosessoinnin pääkohtaa, laatusuunnitelmaa, laatukontrollointia ja laadunparannusta, on toimivan ja kehittyvän yrityksen ylläpitäminen mahdollista.

3 Laatukustannusmittaritietokannan kehitystyö

3.1 Kehitystyön lähtökohdat

PaloDEx Group aloitti nykyaikaisen mallisen laatuseurannan vuonna 2008. Alkuperäisen laatukustannustietokannan oli silloin luonut Jouni Onnela opinnäytetyönä PaloDEx Groupille. Sen tavoitteena oli luoda laatukustannusmittari, jonka avulla voidaan tutkia ja mitata yhtiön tuotannon huonon laadun aiheuttamia kuluja. Onnelan työssä tutkittiin mitattavien kulujen tarpeet ja mahdollisuudet työn toteuttamiseen yrityksen sisällä johtoportaan ja IT-tuen avustuksella. [12, s. 1.] Tuolloin, kuten nykyäänkin, oli tärkeää huomioida kehitettävän sovelluksen loppukäyttäjiltä vaatimat taitotasot. Kuukausittain laadittavan laatukustannusmittarin laatii henkilö PaloDEx Groupin laatuosastolta, joten häneltä ei vaadita hyviä tiedonkäsittelyn ja tietoteknisiä taitoja. Kehitettävän tuotteen tulee näin ollen olla laaja, tarkka ja luotettava mittari ja samalla myös helppokäyttöinen sekä helposti ymmärrettävä henkilölle, joka ei Microsoft Access- ja Microsoft Excel-työkaluja osaa käyttää perustoimintoja enempää.

Nämä asiat tiedostaen Onnela oli työssään päätynyt Access-sovelluksen kehittämiseen IT-osaston kanssa. Siitä voitiin ajaa sovitut tiedot yhteiseksi taulukoksi ja tuoda ne Excel-taulukkoon, josta tulosten lukeminen on helppoa. Suurena osatekijänä alustojen ja esitystavan valinnassa oli se, että työntekijän tietokoneessa oli valmiiksi Microsoftin ohjelmistot, jolloin ohjelmistoista yritykselle ei syntynyt mitään lisäkustannuksia. Onnelan kehitystyön lopussa ja mittarin esittelyssä ja jalkautuksessa pidettiin koulutus- ja tiedotustilaisuus sekä johdolle että laatuosastolle. Tällöin laatupäällikkö ja laatuinsinööri olivat todenneet, että ennen, ilman mittaria, laatukustannusten käsittelyyn ja analysointiin olisi mennyt kokonainen työviikko. Mittarin julkaisun jälkeen, kun kaikki raskas käsittely oli saatu työstymään Access-sovelluksella, käsittely ja analysointi eivät enää vieneet kuin noin kaksi päivää. [12, s. 25.] Lopputulemana Onnela oli kehittänyt siihen hetkeen soveltuvan laatukustannusmittarin, joka nopeutti ja tehosti kustannusdatan käsittelyä huomattavasti. Aivan tyytyväisiä sovellukseen ei vielä johdon puolelta kuitenkaan oltu, sillä käsittelyyn kuului edelleen turhaa käsityötä, mikä olisi hyvä saada toteutettua automaattisesti Access-ajon aikana, mutta sillä hetkellä yrityksellä ei ollut hankkeeseen resursseja ja intressejä [12, s. 29–30].

Viisi vuotta hankkeen lopetuksen jälkeen yrityksessä on käytetty Onnelan kehittämää mittaria kuukausittain, mutta asiat, jotka kehitysvaiheessa jäivät tekemättä, nousivat koko ajan enemmän pintaan ja lopulta hidastivat toimintaa niin paljon, että lopulta laatukustannusmittarin käyttö venyi jo kahdesta kolmeen päivään. Lisäksi viidessä vuodessa yrityksessä on ehtinyt tapahtua uusia tuotejulkaisuja ja suunnanmuutoksia, joita ei ole kannassa otettu huomioon, ja täten tietokanta oli hyvinkin vajavainen ennen kuin aloin tehdä omaa jatkokehitystäni sen parissa. Työn aloituspalaverissa PaloDEx Groupin laatuosaston kesken käytiin läpi mittarin silloinen tilanne ja se, mitä asioita vuosien varrella on tapahtunut koko mittarin käytön kannalta. Lisäksi keskusteltiin, mitä asiakohtia tulisi lähteä kehittämään, jotta mittarin käyttö olisi taas luontevaa ja tehokasta.

Kehityspalaverissa käytiin ensimmäisenä läpi koko laatukustannusmittarin työnkulku ja prosessit. Näin sain itse ensi kosketuksen ongelmiin, jotka minun tulisi seuraavien kuukausien aikana ratkaista. Laatuhenkilöstö nosti edellisestä versiosta esiin seuraavanlaisia ongelmakohtia, joihin toivoi parannuksia.

- Tärkeimpänä ongelmana oli tuoteperheiden puuttuminen kultakin riviltä, eli kustannukselta. Ilman tuoteperhettä ei vanhan Access-sovelluksen luomasta Excel-tiedostosta pystynyt helposti tutkimaan kustannuksia tuotelinjoittain ja sitä kautta kartoittamaan, miten kukin tuotantolinja kehittyy ja edistyy tuottamalla vähemmän laatukustannuksia. Edellisessä versiossa tämä työ jouduttiin tekemään käsin, pääasiassa tutkimalla tuotteen tai osan nimeä ja lisäämällä niitä vastaava tuoteperhe oikeaan sarakkeeseen.

- Vanha tietokanta vaati liian paljon tuotavien taulukoiden esikäsittelyä, kuten otsikoiden uudelleennimeämistä, poistamista, datan seulontaa ja tutkimista. Joidenkin taulukoiden sisältämä data on vuosien varrella muuttanut muotoaan, tai Access-tietokannalle viitepisteinä toimivat otsikot ovat muuttaneet kirjoitusasuaan, jolloin kukin ajettava taulukko tuli käsitellä samaan muotoon kuin alkuperäiseen tietokantaan oli määritelty. Kaikki hidastava esikäsittely tahdottiin minimiin, jotta sitä kautta myös mahdollisuus inhimillisiin erehdyksiin olisi minimissään, ja näin ollen aikaa vieviä jälkikorjauksia ei tarvitsisi tehdä.

Palaverissa esitettiin näiden isojen ongelmien lisäksi toiveita siitä, mihin voisi perehtyä tarvittaessa ja pyrkiä kehittelemään mahdollisia ratkaisuja. Tämän tason ongelmia oli esimerkiksi nimenomaisesti taulukoiden muodon mahdollisten muutosten ennakoimi-

nen. Olisiko mahdollista luoda Accessiin tapa, jolla pienet muutokset lähdetaulukoihin eivät sotkisi koko laatukustannusmittarin suoritusta? Toinen asia, johon toivottiin paneutumista, oli koko Yhdysvaltojen takuukorvaustaulukon sisältö ja sen aiheuttamat ongelmat lopputulostauluun. Yhdysvaltojen takuukorvaustaulukko on aiheuttanut turhia moninkertaisia rivejä lopulliseen taulukkoon ja osittain tyhjiä rivejä, jotka on tullut käsin jälkikäteen täyttää omien tietojen ja taitojen turvin.

3.2 Tietokannan rakenne ja toimintaperiaate

Tietokantasovellus on rakennettu toimimaan perusperiaatteiltaan siten, että ulkopuolisista lähteistä tuodut Excel-taulukot luetaan tietokannan sisälle ja muokataan kaikkien taulukoiden sisältämä data yhtenäiseksi. Tämän jälkeen kaikkien taulujen data yhdistetään yhteiseksi taulukoksi, joka lopulta lähetetään Excel-ohjelmistoon, jossa taulukko on kaikille helposti luettavassa muodossa. Tämän jälkeen dataa voidaan analysoida ja laskea lopullisia kuluja, jotka laaduttomuus on yhtiölle kuukaudessa aiheuttanut. Tietokantasovellus on rakennettu siten, että kaikki tämä tapahtuisi automaattisesti, eli käyttäjän tarvitsee ainoastaan käynnistää tietokannan ajo ja kaikki toiminta tapahtuu automaattisesti Excel-taulukon aukeamiseen saakka.

Kun tietokantasovellusta ja sen osatekijöitä lähdetään purkamaan toiminnallisiin osiin, voidaan tietokannan toiminta hajauttaa kolmeen pääryhmään. Nämä ryhmät, jotka mahdollistavat tietokannan tiedonhallinnan, ovat taulukot, kyselyt ja moduulit. Näiden lisäksi tietokanta sisältää lomakkeen, joka toimii kaiken toiminnan perustana ja kertoo tietokannalle, mitä tehdään missäkin tilanteessa. Näiden kaikkien elementtien yhteistoiminnalla sovellus osaa hakea dataa, muokata sitä ja luoda uuden taulukon. Tämä toiminnallisuuksien yhteistyö tapahtuu edellä mainittujen kolmen pääryhmän avulla. Ensimmäisellä ryhmällä, taulukoilla, on yksiselitteinen, mutta keskeinen tehtävä. Taulukot ovat tietokannan rakenteessa elementtejä, joiden päätehtävänä on tiedon tallentaminen. Muut tietokannan elementit ovat vuorovaikutuksessa taulukoiden kanssa, lisäten, muokaten tai poistaen sisältöä. [13, s. 14.]

Toisena pääryhmänä tietokannan rakenteessa ovat kyselyt. Kyselyitä voidaan käyttää moniin eri tarkoituksiin ja suorittamaan eri funktioita. Yleisesti kyselyjä käytetään tapana paikallistaa dataa tietokantaan määritellyistä taulukoista [13, s. 20]. Tämäntyyppisiä kyselyitä kutsutaan valintakyselyiksi. Toinen kyselytyyppi on toimintokysely, jota voi-

daan käyttää nimensä mukaisesti suorittamaan toiminto datalle. Tällaisia toimintoja ovat esimerkiksi uuden taulukon luonti tai olemassa olevan datan muokkaus. [14.] Tyypillisin, valintakysely, voidaan suorittaa joko hakemalla kaikki data, jonka taulukko sisältää, tai määrittelemällä parametrit, joita tietyn sarakkeen tulee noudattaa, ja hakemalla vain nämä parametrit täyttävät rivit. [13, s. 20.] Pelkällä haetulla datalla ei vielä tehdä mitään, vaan se tulee palauttaa myös jonnekin, kuten erilliseen taulukkoon. Kyselyt suoritetaan SQL-ohjelmointikielellä määritellyn käskyn mukaisesti. Valintakyselyt suoritetaan pääasiassa SELECT- ja FROM-komennoilla, jotka määrittävät, mitä dataa halutaan hakea ja mistä sitä haetaan. Näiden lisäksi haulle voidaan määrittää WHERE-komennolla parametrit, jotka hakutuloksen tulee sisältää. Nämä kolme peruskomentoa mahdollistavat yksinkertaisten SQL-kyselyiden suorittamisen, jossa halutaan etsiä dataa. Tietokannassa ei ole hyödynnetty kuin näitä niin sanottuja valintakyselyitä, sillä monimutkaisemmat datan muokkaukset suoritetaan kolmannen pääryhmän alaisissa moduuleissa.

Moduulit ovat ohjelmia, jotka sisältävät yhden tai useampia Microsoft Visual Basic for Applications (VBA) -ohjelmointikielellä luotuja funktioita eli toimintoja. VBA on Microsoftin luoma ja kehittämä ohjelmointikieli, jolla voidaan luoda ja ohjata Microsoftin ohjelmia. Se on ohjelmointikieli, jota voidaan käyttää kaikissa Microsoft Office -ohjelmistoissa, ja se sisältää satoja eri komentoja valmiiksi, mutta on tästäkin vielä laajennettavissa ulkopuolisilla lisäosilla. [13, s. 30.] Visual Basic -ohjelmoinnilla voidaan helposti suorittaa vaikeitakin toimintoja, minkä takia sen moduulit ovatkin tehokas työkalu muuttuvan datan tulkinnassa ja muokkauksessa. Lisäksi VBA-funktioita voidaan suorittaa suoraan SQL-kyselyn sisällä, mikä mahdollistaa saumattoman yhteistyön kaikkien kolmen pääryhmän välillä ja täten monimutkaisten tiedonhallinnallisten ohjelmien toiminnan.

Seuraavaksi perehdytään tarkemmin PaloDEx Group Oy:n laatukustannusmittarin toiminnallisiin osiin suoritusjärjestyksessä tarkastellen niitä teoreettisesti ja selvittäen niiden käytännön hyödyt.

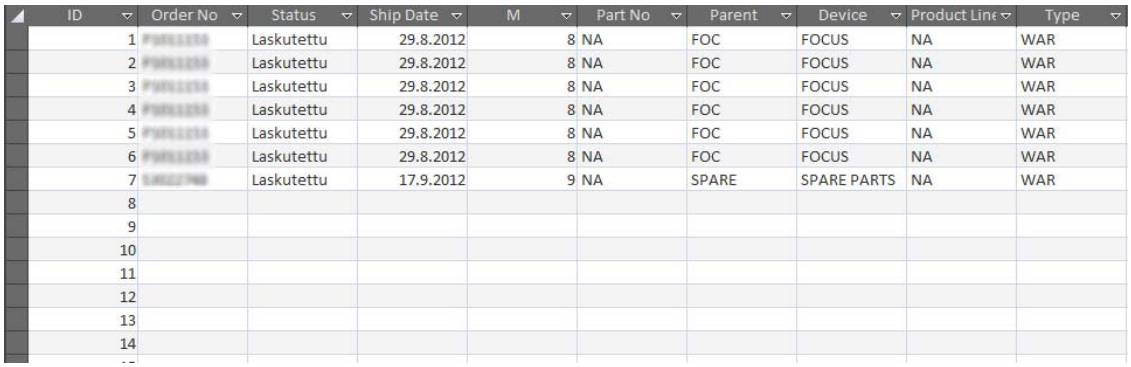
Tietokannan selkärankana toimii lomake, jonka toiminta on verrannollinen moduulien toimintaan. Lomake on ikkuna, joka sisältää ohjauselementtejä ja on joissain tapauksissa keino luoda käyttäjäystävällinen käyttöliittymä helpottamaan datan lisäämistä taulukoihin. Lomakkeen ei kuitenkaan tarvitse olla käyttäjälle näkyvä elementti, vaan kuten kyseisen mittarin kohdalla, tietokannan perusohjain, joka jakaa käskyjä muille

elementeille kertoen, milloin kukin tulee suorittaa. Tämä kaikki tapahtuu, kuten moduulien kohdalla, Visual Basic -ohjelmoinnin keinoin. Ensimmäisenä työnään lomake määrittää, että tietokanta tarvitsee sisälleen ennalta määriteltyihin taulukoihin dataa, jota voidaan ruveta myöhemmissä vaiheissa sulauttamaan ulosannettavaan muotoon.

3.2.1 Taulukot

Tietokannassa on aiemmin esitellyn pääryhmän, taulukot, alle rakennettu valmiiksi taulukkopohjia, jotka vastaavat tietokantaan sisälle tuotavien taulukoiden rakenteita. Tietokantaan tuodaan dataa yhteensä yhdestätoista eri Excel-taulukosta, joten tietokannan sisälle tulee määritellä taulukot, eli niin sanotusti tyhjät datapankit, joihin kaikki tämä tieto sijoitetaan. Jotta tieto osataan sijoittaa suoraan Excel-taulukosta tietokannan taulukoihin, tulee jokaiselle sarakkeelle olla määritelty Excel-taulukkoa vastaavat nimet, eli avaimet, jotka toimivat viitteenä tietoa sijoitettaessa. Tietokantaan sisällytettävässä taulukossa tulee lisäksi aina olla määritelty niin kutsuttu perusavain, joka toimii yksilöllisenä tunnisteena kullekin taulukon riville. Perusavaimet toimivat rivien identiteettinä, ja relaatiotietokantamallisessa toteutuksessa niiden avulla voidaan luoda yhteyksiä taulukoiden välille. [15, s. 9.]

Tässä tapauksessa taulukot eivät kuitenkaan toteuta relaatiotietokannan mallia, jossa taulukoiden välille luodaan yksittäisiä tai moninaisia yhteyksiä. Taulukot varastoivat dataa juuri siinä muodossa ja määrässä, joka niihin lomakkeen käskystä tuodaan jakamatta dataa mitenkään tyypeittäin. Tämä toimintamalli nopeuttaa toimintaa ja helpottaa tietokannan rakentamista, eikä relaatiomallinen toteutus edistäisi tietokannan tarkoitusperää. Näiden asioiden valossa taulukoiden tunnisteet, eli perusavaimet, eivät ole tärkeä osa tietokannan rakennetta. Tästä syystä perusavaimien arvoiksi voidaan määritellä annettavan automaattinen, irrelevantti, juokseva numerointi. Tällaista perusavainta kutsutaan keinoavaimeksi, eli tekniseltä nimeltään surrogaatiksi. Surrogaatit ovat lyhyitä ja muuttumattomia, sillä ne eivät sisällä informaatiota. Taulukoiden koko pysyy täten pienempänä kuin määriteltäessä riveille niin sanotut luonnolliset perusavaimet, jotka vastavuoroisesti sisältävät relevanttia informaatiota rivin sisällöstä. [15, s. 62–63.] Kuvassa 4 nähdään, kuinka taulukko rakentuu tietokannassa ja että sen perusavaimena käytetty ID-sarake sisältää surrogaatteja, sillä tietokannan toiminnallisuuksissa seuraavassa vaiheessa suoritettavien kyselyiden vuoksi haetaan vain tiettyjen sarakkeiden sisältöä eikä koko riviä perusavaimen mukaan.



ID	Order No	Status	Ship Date	M	Part No	Parent	Device	Product Line	Type
1	[illegible]	Laskutettu	29.8.2012	8	NA	FOC	FOCUS	NA	WAR
2	[illegible]	Laskutettu	29.8.2012	8	NA	FOC	FOCUS	NA	WAR
3	[illegible]	Laskutettu	29.8.2012	8	NA	FOC	FOCUS	NA	WAR
4	[illegible]	Laskutettu	29.8.2012	8	NA	FOC	FOCUS	NA	WAR
5	[illegible]	Laskutettu	29.8.2012	8	NA	FOC	FOCUS	NA	WAR
6	[illegible]	Laskutettu	29.8.2012	8	NA	FOC	FOCUS	NA	WAR
7	[illegible]	Laskutettu	17.9.2012	9	NA	SPARE	SPARE PARTS	NA	WAR
8									
9									
10									
11									
12									
13									
14									

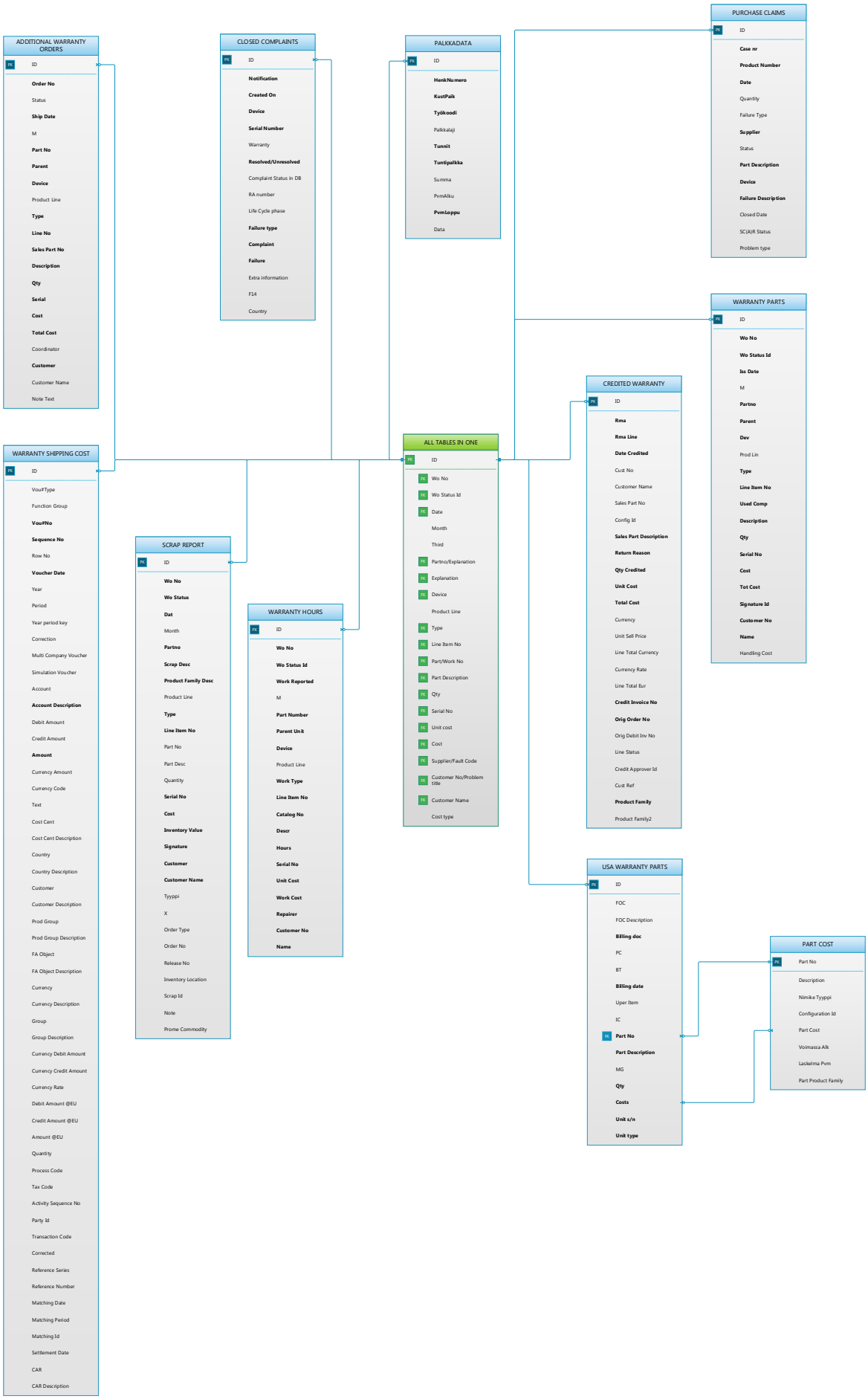
Kuva 4. Ote taulukon rakenteesta tietokannassa.

Tietokantaan tuotavien 11 taulukon lisäksi on tietokantaan määritelty taulukko, johon kaikista muista taulukoista sijoitetaan dataa SQL-hakujen avulla. Tämä taulukko koostuu sisällöstä, jonka katsotaan tukevan laatukustannusten analysointia. Vaikka mainitut 11 taulukkoa sisältävät eri informaatiota, on haettavat datat määritelty siten, että kaikista taulukoista saadaan lähes kaikki informaatio suoraan ja puuttuva tärkeä data voidaan luoda itse tai haun aikana etsiä relaatioita toisesta taulusta, kuten esimerkiksi tietyn osan listahinta. Lopullinen taulukko, johon ulkopuolisista lähteistä saatava data sijoitetaan, sisältää seuraavat sarakkeet, jotka vaaditaan täytettäväksi.

- ID: perusavaimena toimiva sarake, johon luodaan surrogaatteja
- Wo No: työ- tai tilausnumero kululle
- Wo Status Id: työn tilannemääritelmä, esimerkiksi ”Laskutettu”
- Date: päivämäärä, jolloin kulu on kirjattu
- Month: laskutuskuukausi, jonka sisälle kulu sijoittuu
- Third: laskennallinen vuosikolmannes laskutuskuukauden mukaan
- Partno/Explanation: laskutettavan osan tuotenumero tai selitys
- Explanation: lisäselvitys tuotteesta tai viasta
- Device: laite, jonka rakenteeseen kulu ohjautuu
- Product Line: tuotelinja tai minitehdas, joka määräytyy Device-sarakkeen sisällön mukaan VBA-ohjelmoinnin avulla
- Type: kulun tyyppi, esimerkiksi takuukorjaus, uusintatyö jne.
- Part/Wo No: viallisen osan tunniste

- Part Description: viallisen osan täydellinen nimi
- Qty: viallisten osien määrä yhden laskutuksen sisällä
- Serial No: laitteen sarjanumero, josta viallinen osa löytyi
- Unit cost: yhden osan listahinta, saadaan joko suoraan tietokantaan tuoduista taulukoista tai haettua osan tunnisteen mukaan taulukosta, joka sisältää osaluettelon ja hinnaston
- Cost: osan listahinta kerrottuna osien määrällä, eli yhtiöltä laskutettava hinta
- Supplier/Fault Code: viallisen osan toimittaja
- Customer No/Problem title: spesifinen selvitys viasta ja sen vaikutuksista
- Customer Name: viallisen laitteen omistaja
- Cost type: kulun lopullinen laatu; romu, uusintatyö, takuu tai ostoreklamaatiokäsittely.

Kuvassa 5 näkyy koko tietokannan taulukkorakenne ja vaadittavat saraketiedot tietokannan relaatiomallin avulla. Edellä mainittuihin sarakkeisiin tuotava data määräytyy lomakkeen (liite 1) ohjaamien SQL-kyselyiden mukaan riippuen siitä, mitä informaatiota kukin taulukko sisältää. Seuraava prosessi tietokannan käsittelyssä onkin SQL-kyselyiden suorittaminen määrätyssä järjestyksessä.



Kuva 5. Tietokannan entiteettirelaatiomalli.

3.2.2 SQL-kyselyt

Tietokannoissa dataa käsitellään joukko-opillisesti, eli esimerkiksi taulukon käsitetään koostuvan joukosta eri rivejä. Näille joukoille voidaan tietokannassa suorittaa erinäisiä joukko-operaatioita, esimerkiksi valinta, joka vastaa käskyä ”hae”. Joukko-operaatioilla voidaan käsitellä useita taulukkoja yhtäaikaisesti ja myös muokata niitä tarvittaessa. Joukko-oppi onkin perustana tietokantojen käsittelyssä, ja tätä joukko-opillisuutta toteutetaan IBM:n 1970-luvulla kehittelemällä Structured Query Languagella (SQL). SQL-kieli on vakiintunut standardiksi relaatiotietokantojen käsittelijänä, koska se on monipuolinen ja vahva tietokantakieli. SQL-kielen avulla tietokannoille annetaan käskyjä, mitä dataa halutaan hakea, sen sijaan, että käskettäisiin, miten data halutaan haettavan. Tätä toimintatapaa kutsutaan niin sanotusti ”ei-proseduraaliseksi”. Hakujen lisäksi SQL on tehokas kieli muokkaamaan taulukoiden sisältämää dataa. [15, s. 10.]

Kyselyt ovat tärkein lähtökohta toimivan tietokannan rakenteessa, sillä niiden avulla käyttäjälle tuodaan haluttu data luettavaksi. Liitteen 1 mukaisesti lomake alkaa suorittaa kyselyitä, kun kaikki taulukot on tuotu tietokantaan sisälle ulkopuolisista lähteistä. Kyselyiden perusperiaate tietokannan toiminnan kannalta on etsiä taulukoiden sisältä sitä tietoa, mikä laatukustannuksia analysoitaessa on relevanttia, eli dataa, joka sopii edellä mainittuihin sarakkeisiin lopulliseen taulukkoon.

Kun informaatiota halutaan hakea tietyn taulukon sisältä, käytetään jo aikaisemmin esille tuotuja yksinkertaisia valintakyselyitä, eli SELECT (valitse) ja FROM (mistä valitaan). Yksinkertaisin malli SQL-valintakyselylle on ’SELECT * FROM taulun_nimi’, joka tuo kaiken datan, jonka taulun_nimi-taulukko sisältää. SQL-lausetta voidaan laajentaa ja tarkentaa niin laajasti, kuin vain on mahdollista ja järkevää. SELECT-valintakäskyn perään voidaan määritellä, mitä tietoja haetaan sarakkeen otsikkotason perusteella. FROM määrittää taulukon, jossa otsikkotasot sijaitsevat, jotta SQL-kysely osaa kohdistaa haun oikeaan taulukkoon. SQL-kyselyille voidaan myös asettaa ehtoja, joita tuloksen tulee noudattaa. Nämä ehdot määritellään WHERE-käskyn avulla. Käytännön esimerkkinä ’SELECT Date FROM taulun_nimi WHERE Date <> 1.1.2000’. SQL-haku toteutettuna tällä rakenteella tuo näkyviin taulun_nimi-taulukosta päivämäärät, jotka eivät ole 1.1.2000. Ehtojen avulla voidaan siis suodattaa turhaa tai irrelevanttia tietoa taulukoista, jotta käsiteltävä tietomäärä saataisiin pidettyä minimissään ja täten tietokannan kuormitus vähäisenä. [16.] Ehtojen luominen hakuihin on laatukustannusmittarin kannalta elintärkeää, sillä taulukot sisältävät käsittelemättöminä paikoittain paljon

dataa, jota ei haluta lopulliseen taulukkoon mukaan. Ilman ehtolauseiden käyttöä tulisi jälkikäsittelyvaiheeseen niin paljon suoritettavaa ja käsiteltävää, että taulukon nopea viimeistely analysointia varten ei olisi mahdollista. WHERE-ehtolauseella voidaan siis karsia helposti ylimääräinen data pois, mutta sen ei tarvitse aina olla hyvin spesifinen. WHERE-ehtolauseeseen voidaan tietyn arvon sijaan myös sijoittaa ehto NOT NULL, joka määrittää, että rivillä ei saa määritetyssä sarakkeessa olla tyhjää kenttää. Tämäntapaisella toteutuksella voidaan karsia laatukustannusmittarista esimerkiksi kaikki rivit, joilla ei ole kirjauspäivää, tuotenumeroa ja niin edelleen. Kuvassa 6 näkyy esimerkkikoodi taulun SQL-kyselystä. SQL-kyselyssä sarakkeelle pvmalku on määritelty ehto, että se ”ei saa olla tyhjä”, muuten koko riviä ei tuoda tietokantaan sisälle.

```
INSERT INTO [All tables in one] ( ID, [Wo No],
[Wo Status Id], [Date], [Month], Third, [Partno/Explanation],
Explanation, Device, [Product Line], Type, [Line Item No],
[Part/Work No], [Part Description], Qty, [Serial No],
[Unit cost], Cost, [Supplier/Fault Code],
[Customer No/Problem title], [Customer Name], [Cost type] )
SELECT PalkkaData.ID, PalkkaData.HenkNumero AS [Wo No],
"" AS [Wo Status Id], Format([PvmAlku],"Short Date") AS [Date],
Month(Now()) AS [Month], getThird([Month]) AS Third,
PalkkaData.KustPaik AS [Partno/Explanation], "REWORK" AS Explanation,
IIf(IsNull([B]),"REWORK",[B]) AS Device, getProductLine([Device]) AS [Product Line],
"REWORK" AS Type, "" AS [Line Item No], PalkkaData.Työkoodi AS [Part/Work No],
"REWORK" AS [Part description], PalkkaData.Tunnit AS Qty, PalkkaData.Tuntipalkka AS [Serial No],
"45,90" AS [Unit cost], [Qty]*[Unit cost] AS Cost, "" AS [Supplier/Fault code],
"" AS [Customer No/Problem title], "" AS [Customer name], "REWORK" AS [Cost type]
FROM (PalkkaData LEFT JOIN [Cost centers for devices] ON PalkkaData.KustPaik = [Cost centers for devices].A)
INNER JOIN [Cost centers for product lines] ON PalkkaData.KustPaik = [Cost centers for product lines].C
WHERE (((Format([PvmAlku],"Short Date")) Is Not Null) AND ((PalkkaData.Palkkalaji)="113"));
```

Kuva 6. Esimerkki SQL-kyselystä tietokannassa.

WHERE-lauseeseen on esimerkkitapauksessa määritelty myös toinen ehto riville. Lisäehtoja voidaan kirjoittaa WHERE-lauseen sisälle AND-, OR- ja NOT-operaattoreita käyttäen. Tässä tapauksessa, kun halutaan luoda ehdot kahteen erilliseen sarakkeeseen, joiden molempien tulee täyttyä, käytetään AND-operaattoria. AND-operaattorin jälkeinen ehtolause määrittää esimerkkitapauksessa sen, että tietokantaan ei tuoda muita rivejä kuin ne, joiden Palkkalaji-sarakkeeseen on määritelty arvo 113.

SQL-kyselyillä voidaan siis esimerkkitapausta tutkien siististi määrittää, mitä haetaan (SELECT), mistä haetaan (FROM) ja millä ehdoilla (WHERE). Tämän perusrakenteen omaava kysely tuottaisi hakutuloksen ja palauttaisi sen käyttäjälle luettavaksi. Tämä toimintamalli kävisi hyvin, jos tietokantaa käytetään tiettyjen asioiden etsimiseen tai arvojen seuraamiseen. Laatukustannusmittarin tapauksessa asia ei ole niin suoraviivainen. Tietokanta toimii enemmän käsittelijänä ja sisällöntuottajana kuin

tavaravarastona, josta haetaan haluttaessa dataa. Tämän nojalla kaikki suodatettu ja muokattu data täytyy myös tallentaa johonkin, jotta se voidaan jälkeenpäin tuoda ulos tietokannasta käsittelyä ja analysointia varten.

Kyselyn tuloksien tallentaminen toiseen taulukkoon tehdään samassa syklissä itse haun kanssa. Tallennus tapahtuu INSERT-käskyn avulla. Käskyn toiminta on hyvin yksiselitteinen ja nimestä pääteltävissä, kuten muutkin SQL-komennot. INSERT sijoittaa halutun datan haluttuun paikkaan, jossa tämä haluttu paikka määritellään INTO-lisäyksellä. Ilman INTO-jatketta, kysely ei tietäisi, mihin data tulee siirtää. [17, s. 157.] Kuten kuvan 6 esimerkkitapauksessa nähdään, on kyselyn alkuun määritelty INSERT INTO -komento, joka kertoo tietokannalle kyselyä suoritettaessa, että aina, kun kysely paikallistaa ehtoihin sopivan rivin, sen sisältämä data sijoitetaan ”All tables in one” -taulukkoon määrättyihin sarakkeisiin. INSERT INTO -käskyn kanssa tulee ottaa huomioon lähde- ja kohdesarakkeiden määrittelyt. Niiden molempien tulee noudattaa samaa tyyppiä, esimerkiksi molempien tulee olla tekstiä tai numeroita. Jos kohdetaulun saraketyypin on määritelty olevan numero ja arvon maksimipituuden kymmenen merkkiä, ei sinne voida sijoittaa tekstiä, joka on 20 merkkiä pitkä. Sijoitettavan datan tyypin ja pituuden tulee siis noudattaa samoja ehtoja kuin kohdesarakkeessa [17, s. 157].

Ihannetapauksessa INSERT INTO -komento osaa itse päätellä, minkä sarakkeen sisältö sijoitetaan mihinkin sarakkeeseen kohdetaulussa. Tämä onnistuisi ainoastaan siinä tilanteessa, jossa sekä lähdetaulukon sarakkeiden tunnukset, eli otsikot, ovat identtiset kohdetaulukon sarakkeiden tunnisteiden kanssa. Kyseisessä toteutuksessa tämä on kuitenkin mahdottomuus, sillä taulukoita tuodaan useista lähteistä ja ne sisältävät useaa eri dataa, joten niiden rakenne ja tunnisteet vaihtelevat huomattavasti. Jotta INSERT INTO -komento osaisi kyselyä suoritettaessa yhdistää lähdetaulukon saraketiedon kohdetaulukon sarakkeeseen, tulee SELECT-komennon yhteydessä määritellä, mitä kukin lähdetaulukon sarake vastaa kohdetaulukossa, kun tunnisteet ovat täysin erilaiset keskenään.

Käytetään yhä kuvaa 6 esimerkkinä tästä toteutuksesta. INSERT INTO -komennon perään määritellyt sarakenimet toimivat viitteinä kohdetaulun, tässä tapauksessa ”All tables in one”, sarakkeisiin. Näihin viitteisiin yhdistetään kukin lähdetaulukon sarake rakenteella [sarakkeen_nimi] AS [kohteen_nimi]. AS-syntaksi muuttaa sen kohdistamalle sarakkeelle väliaikaisesti nimen joksikin muuksi [18]. Joissain

tapauksissa tätä menetelmää käytetään monimutkaisten sarakenimien yksinkertaistamiseen, mutta esimerkin tapauksessa AS-syntaksia käytetään sarakkeiden muokkaamiseen vastaamaan kohdetaulukon sarakkeita. Käytännön muotoon kirjoitettuna esimerkin mukainen käsky ”SELECT PalkkaData.Henknumero AS [Wo No]” olisi ”Valitse taulukosta PalkkaData sarake Henknumero ja käsittele se niin kuin sarakkeen nimi olisi Wo no”. Kun kyselyn kautta on määritelty, että sarake Henknumero käsitellään Wo no -sarakkeena, se osaa yhdistää sarakkeen sisällön INSERT INTO -komennolla määritetyn taulukon samaiseen Wo no -sarakkeeseen.

INSERT INTO -komennolla pystytään mahdollistamaan laatukustannusmittarin vaatimus, että suodatettu ja modifioitu data sijoitetaan yhteiseen taulukkoon, joka tuodaan tietokannasta ulos. SELECT-komennolla saadaan määriteltyä, mitkä taulukon osat käsitellään ja millä tunnisteella niitä tulee käsitellä. FROM-komento kertoo yksinkertaisesti, mistä kaikki tämä data haetaan, ja WHERE-komento täydentää FROM-komentoa määrittäessään ehtoja niille riveille, joita määritetystä lähteestä haetaan. Näillä SQL-kyselyiden peruspilareilla saadaan aikaan monimutkaisia ja monimuotoisia kyselyitä, joilla mahdollistetaan laatukustannusmittarin kokoaminen useista taulukoista yhdeksi isoksi taulukoksi, josta voidaan helposti analysoida kuukauden kulut eri osa-alueittain. Joissain tapauksissa kuitenkaan pelkkä lähdedatan käyttö ei riitä, vaan tietoa tarvitaan enemmän, mutta sitä ei ole vielä olemassa. Tällaisissa tapauksissa ei voida yhdistellä useamman taulukon dataa kyselyyn, vaan sisältöä pitää luoda tarpeen mukaan. Koska kyseessä on automatisoitu tietokantasovellus, on vaateena, että tämä sisällön luominen tapahtuu myös automatisoidusti. Tätä SQL ei komentokielenä kykene suorittamaan, joten sen rinnalle on otettava käyttöön toinen keino luoda sisältöä. Sisältö luodaan kyseisessä toteutuksessa aikaisemmin esiteltyjen moduulien avulla.

3.2.3 Moduulit

Moduulit ovat, kuten aiemmin esitetty, kooste määrityksiä, prosesseja ja tiedotuksia, jotka on pakattu yhdeksi kokonaisuudeksi. Moduulit koostuvat VBA-ohjelmointikielellä kirjoitetuista komennoista ja funktioista, ja niillä tuodaan tietokantaan lisää toiminnallisuutta. VBA-ohjelmointikielellä on helppoa luoda toiminnallisuuksia niin tietokantaan kuin kaikkiin Microsoft Office -ohjelmistoihinkin. Koska VBA voidaan käsittää hyvinkin vahvaksi ohjelmointikieleksi, sillä voidaan luoda lähes mitä tahansa toiminnallisuuksia, niin tietokantaan kuin Exceliin. Office-paketin ohjelmistoissa VBA voidaan valjastaa

käyttöön makrojen muodossa, mutta tietokantojen moduulin sisällä VBA-ohjelmointia voidaan käyttää paljon monimuotoisemmin uuden luomiseen ja datan käsittelyyn. Moduulien sisältö koostuu useista funktioista eli toiminnallisuuksista, jotka on luotu helpottamaan ja parantamaan tietokannan työskentelyä. Tietokannan työskentelyä helpottaa myös se tosiasia, että moduulien sisältämät funktiot toimivat täysin yhteistyössä SQL-kyselyiden kanssa. Jos SQL-kyselyllä ei saada kerättyä kaikkea tarvittavaa dataa, joka lopputulokseen vaaditaan, voidaan kyselyllä määrätyn sarakkeen kohdalla viitata moduulien sisällä määriteltyyn funktioon. Tämä funktio toimii tällöin sen ohjelmoinnin mukaisesti ja palauttaa tuloksen SQL-kyselylle. Tärkein ominaisuus tässä moduulien ja kyselyiden välisessä yhteistyössä on, että moduulien sisäisillä ohjelmoinneilla voidaan hakea dataa suoraan kyseisen kyselyn sillä hetkellä käsittelyssä olevan rivin muista osista. Moduulien sisäisellä ohjelmoinnilla voidaan siis luoda SQL-kyselyn aikana taulukkoon joko muuttumatonta dataa tai dataa, joka määräytyy taulukon muiden sarakkeiden sisällön mukaan.

Esimerkiksi Visual Basic -ohjelmoinnin mahdollistamasta lisäsisällöstä SQL-kyselyllä voidaan ottaa skenaario, jossa tuotettavan taulukon riviin halutaan saada tulostettua päivämäärän perusteella se vuoden kolmannes, jolle tapahtuma sijoittuu. Näin jälkikäteen voidaan esimerkiksi seurata laatukustannusten kehitystä vuosikolmanneksien välillä. Tietokannassa tämä toteutus on tehty kuvan 7 mukaisesti.

SQL -KYSELY

```
INSERT INTO [All tables in one]
( ID, [Wo No], [Wo Status Id], [Date], [Month], Third, [Partno/Explanation], Explanation, Device, [Product Line],
Type, [Line Item No], [Part/Work No], [Part Description], Qty, [Serial No], [Unit cost], Cost, [Supplier/Fault Code],
[Customer No/Problem title], [Customer Name], [Cost type] )

SELECT
[WARRANTY HOURS].ID, [WARRANTY HOURS].[Wo No],
[WARRANTY HOURS].[Wo Status Id], [WARRANTY HOURS].[Work Reported] AS [Date],
Month(Now()) AS [Month], getThird([Month]) AS Third,
[WARRANTY HOURS].[Part Number] AS [Partno/Explanation],
[WARRANTY HOURS].[Parent Unit] AS Explanation, [WARRANTY HOURS].
Device AS Device, getProductLine([Device]) AS [Product Line], [WARRANTY HOURS].[Work Type] AS Type,
[WARRANTY HOURS].[Line Item No], [WARRANTY HOURS].[Catalog No] AS [Part/Work No],
[WARRANTY HOURS].Descr AS [Part Description], [WARRANTY HOURS].Hours AS Qty,
[WARRANTY HOURS].[Serial No], [WARRANTY HOURS].[Unit Cost], [WARRANTY HOURS].[Work Cost] AS Cost,
[WARRANTY HOURS].Repairer AS [Supplier/Fault Code], [WARRANTY HOURS].[Customer No] AS [Customer No/Problem title],
[WARRANTY HOURS].Name AS [Customer Name], "WARRANTY" AS [Cost type]

FROM [WARRANTY HOURS]

WHERE ((([WARRANTY HOURS].[Work Reported]) Is Not Null));
```

VBA -FUNKTIO

```
Function getThird(Month)
If Month > 8 Then
  getThird = 3
ElseIf Month > 4 Then
  getThird = 2
Else
  getThird = 1
End If
End Function
```

Kuva 7. Visual Basic -ohjelmointi ja sen käyttämiseen viittaaminen SQL-kyselyssä.

Annetun esimerkin mukaan nähdään, kuinka SQL-kyselyssä lihavoidussa osiossa haetaan dataa ”Third”-otsikon alaiseen sarakkeeseen ”getThird”-funktion avulla käyttäen hyväksi dataa edellisessä sarakkeessa ”Month”-otsikon alla. Moduulin sisälle on ohjelmoitu funktio ”getThird” Month-muuttujan mukaan, johon edellä viitattiin. Tämä laskee vuoden kolmanneksen kyseisen rivin kuukauden eli ”Month”-sarakkeen datan mukaan ja tämän jälkeen sijoittaa tuloksen taulukkoon ”Third”-otsikon alle. SQL-kyselyn ja Visual Basicin yhteistoiminta näin ollen mahdollistaa laajemman datan käsittelyn ja tuonnin lopulliseen taulukkoon kuin pelkästään SQL:n avulla lähteistä hakemalla.

Ilman moduuleja ja niiden sisältämiä VBA-ohjelmointeja olisi tietokannan sisältö huomattavan paljon vaikeampi luoda ja saavuttaa. Kuvan 7 esimerkkitapaus on hyvin yksinkertainen, mutta toimiva ja tarpeellinen. Koska esimerkki viittaa arvoon ”Month”, jonka kysely luo itse käyttöjärjestelmän päivämäärän mukaan, on VBA-funktion helppo

laskea siitä aina oikea vuosikolmannes. Moduuleihin viittaaminen SQL-kyselyssä ei kuitenkaan aina vaadi oikeaa viittauskohdetta tai dataa itse viittauskohteessa. Tietenkään funktio ei osaa luoda oikeaa sisältöä, jos sille ei anneta tietoa, mitä käyttää viitteenä. Tässä tapauksessa funktio tuottaa tyhjää sarakkeeseen ja SQL voi jatkaa kyselyn suorittamista. Moduulien ja kyselyiden yhteistyö on siis myös hyvin vikasietoista, sillä tyhjän datan luominen ja tyhjät viitteet eivät ole ongelma eivätkä näin aiheuta virheilmoituksia, mikä puolestaan nopeuttaa tietokannan toimintaa. Jos data, johon viitataan, haetaan suoraan tietokantaan tuoduista taulukoista, voi niissä olla kuukausittain paljon eroja ja erilaista sisältöä tietyn sarakkeen sisällä. Tämä johtaa usein tilanteeseen, jossa moduulien avulla luotava data ei ole aina 100-prosenttisen kattava, sillä VBA-funktioon tulisi tässä tapauksessa pystyä ohjelmoimaan viittaussarakkeen kaikki mahdolliset sisältömahdollisuudet. Usein siis taulukoiden epämääräinen data aiheuttaa joitakin tyhjiä kenttiä lopullisessa taulukossa, mutta niissä tapauksissa tyhjien kenttien olemassaolo ei ole olennaista.

Esimerkki tämäntasoisista toiminnoista on aikaisemmassa versiossa puuttuvaksi jäänyt laitteiden tuoteperheiden määrittely. Suurimpana ongelmana vanhassa tietokannassa oli se tosiasia, että tietokantaan tuotavista taulukoista haettavalle datalle ei saatu määriteltyä tuotelinjaa. Tuotelinjojen avulla lopullisesta taulukosta saataisiin helposti jaoteltua kustannukset niiden mukaan, jolloin pystytään paremmin näkemään, mikä tehtaan tuotantolinjoista aiheuttaa paljon kuluja, ja täten aloittamaan toimenpiteitä sen parantamiseksi.

Tietokannassa toiminto luotiin käyttäen hyväksi jo toisaalla hyväksi havaittua moduulien ja kyselyiden yhteistyötä. Kuvan 7 esimerkissä näkyy, kuinka kyselyn SELECT-lauseessa viitataan ”getThird”-funktion lisäksi toiseen VBA-funktioon, ”getProductline”. Funktio käyttää viitearvonaan laitteen nimeä, jonka käyttö tekee tuotelinjan määrittämisestä helppoa. Jokainen tuote kuuluu johonkin tiettyyn tuoteperheeseen, jolloin tuoteperhe saadaan pääteltyä suoraan taulukoista löytyvästä laitteen nimestä. Moduuliin ohjelmoidun funktion haluttiin etsivän tiettyjä merkkijonoja tuotenimen sarakkeesta, jota funktio vertaa sen sisään määriteltyihin tapauksiin. Tämä toimintatapa mahdollistettiin käyttämällä VBA-ohjelmoinnissa ehtolauserakennetta.

Ehtolausekkeilla tutkitaan ja valitaan toimintatapa sen perusteella, mitä ehtolauseeseen tuodaan. Laatukustannusmittarin tapauksessa, kun halutaan funktion tutkivan lähdedataa, on If...Then-rakenteinen ehtolause ideaali. If...Then-ehtolauserakenne tutkii, sisäl-

tääkö sen sisään määritelty muuttuja annetun arvon. Ehtolause tuottaa aina määreisen vastauksen tosi tai epätosi. Ehtolauseet eivät tuota epämääräisiä vastauksia, minkä takia niiden käyttö datan tulkinnassa on tehokasta. If...Then-ehtolauseet mahdollistavat useiden ehtojen asettamisen ElseIf- ja Or-jatkofunktioilla. [19.] Or-jatkofunktion käyttö If...Then-ehtolauseessa antaa useita eri vaihtoehtoja ehdon suorittamiseen. Tällä toimintatavalla voidaan luoda yksinkertaisia hakukoneita moduulien ja kyselyiden välille. Tämä toimintatapa mahdollisti minitehtaiden määrityksessä ehtolauseista koostetun hakukoneen luomisen. Funktio toteutettiin niin, että jokainen aktiivinen tuoteperhe määritettiin omiksi ehtolauseiksi. Näiden ehtolauseiden sisälle on annettu ehtoja käyttäen Like-operaattoria. Like-operaattori etsii solun sisältämästä merkkijonosta määriteltyjä merkkijonojen osia. Määritelty merkkijono voi olla kuinka pitkä tahansa, mutta tuoteperheen määrityksessä tärkein ominaisuus on, että Like-operaattori osaa lukea löysästi solun sisältämän merkkijonon, jolloin kirjoitusmuodoilla ei olisi suurta vaikutusta lopputulokseen. Kuvassa 8 on esitetty esimerkki tietokannassa suoritettavasta Like-operaattorin sisältämästä ehtolauseesta.

SQL-KYSELY

```
INSERT INTO [All tables in one]
( ID, [Wo No], [Wo Status Id], [Date], [Month], Third, [Partno/Explanation], Explanation, Device, [Product Line],
Type, [Line Item No], [Part/Work No], [Part Description], Qty, [Serial No], [Unit cost], Cost, [Supplier/Fault Code],
[Customer No/Problem title], [Customer Name], [Cost type] )

SELECT
[WARRANTY HOURS].ID, [WARRANTY HOURS].[Wo No],
[WARRANTY HOURS].[Wo Status Id], [WARRANTY HOURS].[Work Reported] AS [Date],
Month(Now()) AS [Month], getThird([Month]) AS Third,
[WARRANTY HOURS].[Part Number] AS [Partno/Explanation],
[WARRANTY HOURS].[Parent Unit] AS Explanation, [WARRANTY HOURS].
Device AS Device, getProductLine([Device]) AS [Product Line], [WARRANTY HOURS].[Work Type] AS Type,
[WARRANTY HOURS].[Line Item No], [WARRANTY HOURS].[Catalog No] AS [Part/Work No],
[WARRANTY HOURS].Descr AS [Part Description], [WARRANTY HOURS].Hours AS Qty,
[WARRANTY HOURS].[Serial No], [WARRANTY HOURS].[Unit Cost], [WARRANTY HOURS].[Work Cost] AS Cost,
[WARRANTY HOURS].Repairer AS [Supplier/Fault Code], [WARRANTY HOURS].[Customer No] AS [Customer No/Problem title],
[WARRANTY HOURS].Name AS [Customer Name], "WARRANTY" AS [Cost type]

FROM [WARRANTY HOURS]

WHERE ((([WARRANTY HOURS].[Work Reported]) Is Not Null));
```

VBA-FUNKTIO

```
Function getProductLine(product As String)
Dim temppi As String
Dim Device As String

temppi = ""

If product Like "*OP30*" Or _
product Like "*GXDP*300*" Or _
product Like "*PAN*EXAM*" Or _
product Like "*NOVUS*" Then
temppi = "ELEO"
End If

getProductLine = temppi
End Function
```

Kuva 8. Tietokannan automaattinen sisällöntuotanto Visual Basic -Like-operaattorin avulla.

Liitteessä 2 on esitetty kuvaa 8 tarkemmin tuoteperheen määrittely yksinkertaisten merkkijonojen avulla, joita funktio vertaa taulukon sisältämään dataan. Jos taulukon Device-sarake sisältää tuotteen ”GXDP-300”, se voidaan kirjoittaa oikean kirjoitusasun lisäksi ”GXDP300” tai ”GXDP 300”. Näin tarkkoja määritelmiä ei kannata Like-operaattorin yhteyteen määritellä, sillä tällöin Visual Basic -ohjelmointia tulisi suuri määrä jokaista tuoteperhettä kohden, kun kaikki mahdolliset kirjoitusasut tulisi määritellä erikseen. Tämä ei myöskään tukisi sitä, että SQL:n tulkitsemien taulukoiden luoja olisi tehnyt itse kirjoitusvirheitä. Like-operaattoriin yhdistettävien merkkijonomääritelmien siis tulee olla spesifisiä, mutta silti antaa mahdollisuus useille kirjoitusmuodoille. Tämä saavutetaan liitteen 2 ehtolauserakenteessa näkyvällä merkkijono-osasten erittelyllä *-merkkiä käyttäen. Merkinnällä * VBA-ohjelmoinnissa kuvataan ”mitä tahansa merkkiä tai merkin puutetta”. Tässä tapauksessa määriteltäessä Like-operaattorille merkkijono ”GXDP*300” saadaan katettua kaikki edellä mainitut kirjoitusmuodot yhdellä rivillä.

Valmiiden If...Then-ehtolauseiden järjestäminen kokonaisuudeksi on tarkka ja kriittinen prosessi. Lauseiden sijainti funktiossa määrää, missä vaiheessa suoritetta se luetaan. Vastaavuuden löydyttyä funktio ei kuitenkaan pysähdy, vaan suorittaa kaikki ehtolauseet ennen arvon palautusta. Tämä johtaa siihen, että joissain tapauksissa sarakkeen arvo täyttää useamman ehtolauseen vaatimukset ja siitä huolimatta funktion tulee osata tuottaa oikea tuoteperhe laitteelle.

Tästä tapauksesta on esimerkkinä merkkijono CRANEX, joka esiintyy laitenimissä CRANEX, CRANEX D ja CRANEX 3D, jotka kaikki kuuluvat eri tuoteperheeseen. If...Then-ehtolause suoritetaan ylhäältä alas tarkkaillen, löytyykö sen sisältämille argumenteille vastaavuutta ja kirjoittaa määritellyn merkkijonon arvon muuttujalle ”temppi”. Tuoteperheet on tästä syystä funktiossa järjestelty siten, että ehtolauseet siirtyvät yksinkertaisista monimutkaisempiin, eli esimerkin mukaisesti ensin etsitään merkkijonoa CRANEX, sitten CRANEX+D ja viimeisenä CRANEX+3D. Tämä takaa sen, että virhemarginaali on häviävän pieni samantapaisten laitenimien tutkinnassa. Jos ehtolauseet olisivat päinvastoin, eli pelkän merkkijonon CRANEX-haku olisi viimeisenä, kaikkien CRANEX-merkkijonon sisältävien tuotteiden luettaisiin kuuluvan samaan tuoteperheeseen CLASSIC. Funktion suoritusjärjestys ja sen loppuun asti suorittaminen on siis otettava huomioon vastaavia funktioita tehtäessä.

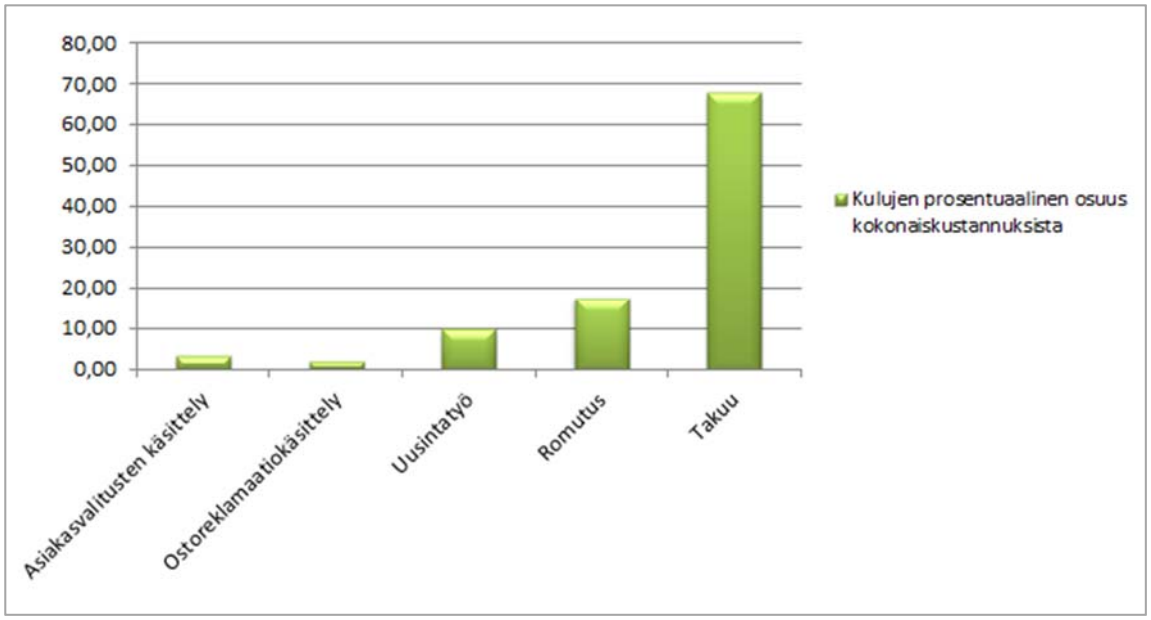
3.2.4 Raportin luonti

Kun tietokanta on saanut kyselyiden kautta siirrettyä ja luotua dataa yhteen yhteiseen taulukkoon, se tulee luvun 2.3 teorian mukaan siirtää helposti ymmärrettävään muotoon, jotta sen analysointi on mielekästä. Koska data on varastoitu tietokantaan taulukon muodossa, se on järkevintä ja helpointa myös tulostaa tietovastaavalle helposti ymmärrettävässä taulukkomuodossa. Taulukot ovat helppoja käsitellä ja jaotella, joten datan tutkimisessa niiden käyttö on tehokasta.

Liitteen 1 sisällöstä näkyy, että kun lomake on ohjannut kaikki kyselyt suoritetuiksi, se siirtyy datan lähetykseen. Tämä prosessi, kuten muutkin lomakkeen ohjaamat toiminnot, tapahtuvat VBA-ohjelmoinnin avulla. Taulukon lähetystä hallitsee komento TransferSpreadsheet, joka hyvin yksiselitteisesti määrittää, että tietokannan halutaan siirtävän taulukon sivu toisaalle. Samaa käskyä voidaan VBA-ohjelmoinnissa lähetyksen lisäksi käyttää datan tuontiin, kuten liitteen 1 ensimmäisessä osiossa nähdään. Tiedon siirron suunta määritellään pääohjaimen, Transferspreadsheet, jälkeen joko AcImport (tuonti) tai AcExport (vienti). Kun tiedonsiirron suunta on selvitetty tietokannalle, tulee vielä määrittää, mitä siirretään. Esimerkkitapauksen mukaan tietokannalle määritetään muoto, johon siirrettävä data sijoitetaan. Jos haluttu muoto on Excel-taulukko, määritetään tietokannalle, että data halutaan viedä Excel-taulukkomuotoon. [20.] Tämä tehdään komennolla acSpreadsheetTypeExcel. Jos käytössä olisi Lotus-järjestelmä, johon data halutaan viedä, muutetaan komentoa vain vaihtamalla kohdealusta Lotukseen, acSpreadsheetTypeLotus. Koska esimerkiksi Excel-ohjelmistosta on useampia eri versioita, voidaan VBA-komentoon määritellä myös, mitä ohjelmistoversiota vastaavassa muodossa taulukko tallennetaan. Tämä määritellään edellisen komennon perään numerolla, joka vastaa ohjelmiston versiointia. [21.] Kyseisessä toteutuksessa käytetty acSpreadsheetTypeExcel12Xml viittaa Excel 2010 -muotoon, jotta tuotu data vastaisi toiminnoiltaan ja mahdollisuuksiltaan nykyaikaisia ja käytössä olevia työkaluja. Kun muotoseikat on määritelty, tulee vielä ilmoittaa, minkä tietokannan osion halutaan siirtyvän Excel-muotoon. Tässä tapauksessa käytetään ”All in one” -taulukkoa, johon aikaisemmissa vaiheissa käsitelty data on luotu.

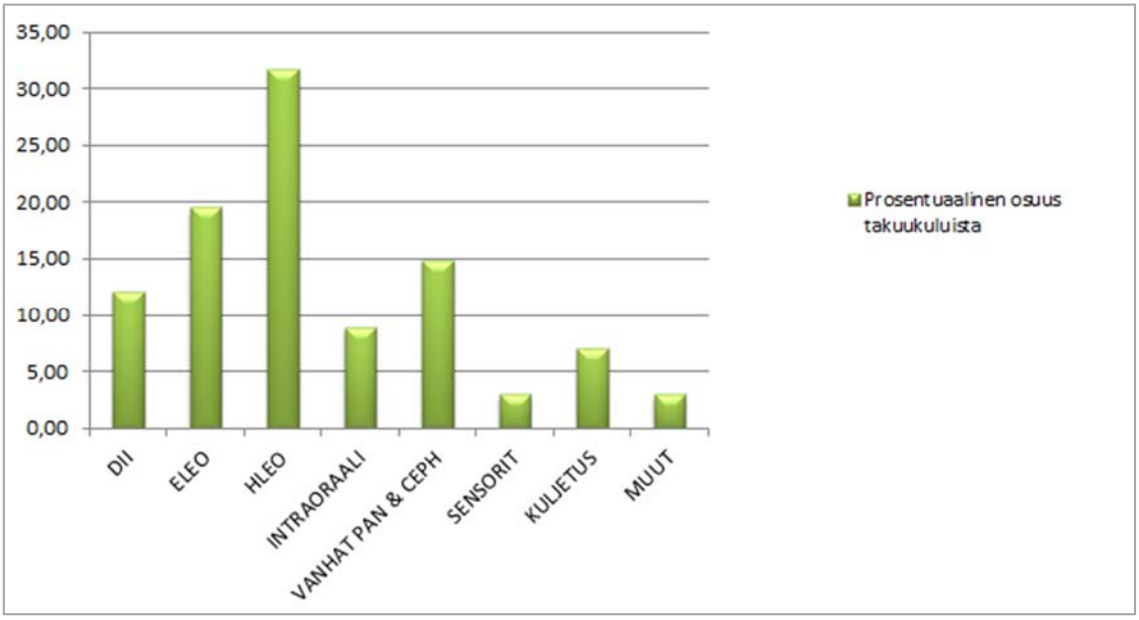
Kun tietokannan sisäinen taulukko on siirretty ulkoiseen muotoon, voidaan dataa alkaa vihdoin analysoida. Kuten laatukustannuslaskennan teoriassa todettiin, pyritään laatukustannuslaskennan analyysilla paikantamaan suurimmat kulujen aiheuttajat ja löytämään niille perimmäiset syyt. Taulukoiden tulee myös sisältää laatukustannusten konk-

retia, yleisimmin rahan muodossa esitettynä. Koko luvussa 3.2 on esitelty, kuinka PaloDEx Groupin laatukustannusmittari rakentuu ja käytännössä käsittelee datan. Kun tietokannan suorite on saatu liitteen 1 lomakkeen avulla valmiiksi, saadaan käsittelyyn tarkka raportti, jota analysoimalla pystytään näkemään kuukausittaiset kuluerät ja suurimmat ongelmakohteet. Esimerkkitapauksena raportin analysoinnista on kuvaan 9 kuvattu PaloDEx Groupin maaliskuun 2013 laatukustannuslaskennan tuottama data pylväsmallisena esityksenä, josta voidaan helposti nähdä takuukustannusten olevan yrityksen suurin laatukustannuksia tuottava osa-alue.



Kuva 9. PaloDEx Group Oy:n laatukustannusmittarin laskennan tulos maaliskuussa 2013.

Kun halutaan päästä takuukustannusten määrän juurille, voidaan mittarin tuloksia ja kustannuksia järjestää tuotantolinjoittain, kuten kuvassa 10 on esitetty.



Kuva 10. Takuukulujen lajittelu tuotantolinjoittain.

Kuvasta 10 voidaan analysoida suurimpien takuukustannusmäärien johtuvan HLEO-linjan tuotteista ja toiminnasta. Tämän perusteella voidaan alkaa tutkia HLEO-linjan tuottamien takuukustannusten aiheuttajia tarkemmin ja lopulta hakeutua ongelmien perimmäisten syiden jäljille luvussa 2.3 esiteltyjen viiden miksi-kysymyksen avulla.

4 Yhteenveto

Insinöörityössä selvitettiin yrityksen suhde laatuun ja laaduttoman tuotannon kustannuksiin. Lisäksi työssä perehdyttiin PaloDEx Group Oy:n laatukustannusjärjestelmän rakenteeseen ja sen kehitykseen vastaamaan nykyisiä ja tulevia tarpeita. Tavoitteena työssä oli selvittää, millä osa-alueilla aiempi laatukustannusmittari toimi vaillinaisesti, ja korjata nämä ongelma-alueet ja luoda uusia toiminnallisuuksia mittariin mahdollistamaan sen tehokkaampi käyttö ja poistamaan työprosesseja laatuhenkilöstöltä.

Toiminnallisen työstön ja useiden prototyyppivaiheiden jälkeen saatiin helmikuussa 2013 laatukustannusmittarista täysin toiminnallinen versio, jolla pystyttiin luomaan tarvittava data nopeasti ja helposti. Edellisen version julkaisussa todettiin, että laatukustannusten laskeminen ja käsittely ja analysointi saatiin nopeutettua useammasta päivästä yhdestä kahteen päivään. Helmikuun 2013 mittariversiolla saatiin kuitenkin tarkemman määrittelyn ja tietokannan automatisoinnin avulla pienennettyä mit-

taus-, käsittely- ja analysointiaika noin tuntiin. Ongelmakohdat näin ollen saatiin selvitettyä onnistuneesti, ja mittari saatiin otettua yritykseen käyttöön sulavasti.

Mittarin luotettavuus pysyi ennallaan ja jopa parani, kun työntekijöiden tekemät välivaiheet vähentyivät. Data, joka mittariin tuodaan, on jo valmiiksi luotettavasta lähteestä tuotettua, joten väärän informaation mahdollisuus on hyvin pieni. Laatukustannuskohteiden selvitys ja taloudellinen menetys näillä alueilla saadaan analysoitua nopeasti ja luotettavasti, mikä edesauttaa yrityksessä entistä nopeampaa reagointia laatuongelmiin ja niiden aiheuttamiin kustannuksiin. Koska edellinen mittariversio oli Jouni Onnelan ja yrityksen IT-osaston tuotos, ei laatuhenkilöstöllä ollut tietoa tietokannan toiminnasta ja sen päivitys jäikin IT-osaston vastuulle. Päivitystä ei kuitenkaan jälkeenpäin tehty, ja tietokannan monimutkaisuuden takia kukaan ei sitä osannut lopulta tehdä. Suurin hyöty tästä insinöörityöstä yritykselle ja etenkin laatuosastolle olikin mittarin päivitys toimivalle tasolle ja sen yksinkertaistaminen sekä ohjeistaminen, jotta laatuhenkilöstö pystyisi tekemään päivitystä ja korjausta. Kun laatuosasto jälleen hallitsee kustannusmittauksen prosessin ja työkalut, paranee koko yrityksen ja etenkin johdon luottamus osastoon.

Nykyisellään mittari ei kestä kauan, sillä laaduttomuutta ja sen laskentaa pyritään parantamaan jatkuvasti organisaation ja organisaatiorakenteen kansainvälistyessä. Tämä vaatii useampien lähteiden tulkintaa ja suurempien massojen käsittelyä, ja tästä syystä laatukustannusmittarin muoto elää seuraavien vuosien aikana. Kun laaduttomuuden laskenta ja tietojen käsittely on laatuorganisaation vastuulla ja nykyisellään myös tietokannan toiminnan päivitys on mahdollistettu, nämä muutokset eivät välittömästi aiheuta ongelmia organisaatiossa ja laatukustannusmittauksessa. Työ mittarin parissa ei siis ole vielä ohi kehittäjän tai laatuorganisaationkaan osalta, vaan vaatimusten ja odotusten kasvaessa laatukustannusmittaus pyritään pitämään ajan tasalla ja tehokkaana.

Lähteet

1 PaloDEx Group. Verkkodokumentti. PaloDEx Group Oy. <www.palodexgroup.com>. Luettu 13.3.2013.

2 Järvinen, P., Lemetti, P., Virtanen, T. 2001. Laatukustannuslaskenta: käyttötarkoitus ja menetelmät. Espoo: Otamedia.

3 Campanella, Jack. 1999. Principles of Quality Costs: Principles, Implementation and Use. Milwaukee: ASQ Quality Press.

4 Wood, Douglas. 2007. The Executive Guide to Understanding and Implementing Quality Cost Programs. Milwaukee: ASQ Quality Press.

5 Pareto Principle. Verkkodokumentti. Pinnicle Management. <http://www.pinnicle.com/Articles/Pareto_Principle/pareto_principle.html>. Luettu 6.4.2013.

6 Atkinson, John. Hohner, Gregory. Mundt, Barry. Troxel, Richard. Winchell, William. 1991. Current trends in cost of quality: Linking the cost of quality and continuous improvement. Montvale, NJ: National association of accountants.

7 Laukkanen, Tero. 2011. Automaatio- ja instrumentointiratkaisujen vaikutus nykyaikaisten työkoneiden tuotannon kustannuksiin. Diplomityö. Tampereen teknillinen yliopisto.

8 Dale, Barrie. 1999. Managing Quality. Oxford: Blackwell Publishing.

9 Laatukustannukset, eli laatuun ja laaduttomuuteen liittyvät kustannukset. Verkkodokumentti. Arvio. <http://www.arvio.fi/artikkelit_laatukustannukset.html>. Luettu 1.4.2013.

10 Root Cause Analysis Using Five Whys. 2008. Verkkodokumentti. NHS Institute for Innovation and Improvement. <http://www.institute.nhs.uk/quality_and_service_improvement_tools/quality_and_service_improvement_tools/identifying_problems_-_root_cause_analysis_using5_whys.html>. Luettu 8.4.2013.

11 Joseph Juran. Verkkodokumentti. Quality Gurus. <http://www.qualitygurus.com/gurus/list-of-gurus/joseph-juran/>. Luettu 24.4.2013.

12 Onnela, Jouni. 2008. Laaduttomuuskulumittarin kehittäminen ja jalkautus PaloDEx Group Oy:ssä. Opinnäytetyö. Lahden Ammattikorkeakoulu.

13 Lambert, Steve. 2007. Microsoft Office Access 2007 (orig. Microsoft Access 2007 Step by Step). Helsinki: Readme.fi.

14 Database basics. Verkkodokumentti. Microsoft. <http://office.microsoft.com/en-us/access-help/database-basics-HA010064450.aspx#BMtables>. Luettu 7.3.2013.

15 Hovi, Ari. Huotari, Jouni. Lahdenmäki, Tapio. 2003. Tietokantojen suunnittelu & indeksointi. Jyväskylä: Docendo Finland.

16 SQL: WHERE Clause. Verkkodokumentti. Tech on the net. <http://www.techonthenet.com/sql/where.php>. Luettu 7.3.2013.

17 Stephens, Ryan. Plew, Ronald. Morgan, Bryan. Perkins, Jeff. 1999. SQL Tietokantaohjelmointi (orig. Teach yourself SQL in 21 Days). Helsinki: IT Press.

18 SQL Alias. Verkkodokumentti. W3Schools. <http://www.w3schools.com/sql/sql_alias.asp>. Luettu 7.3.2013.

19 Halvorson, Michael. 2008. Visual Basic 2008 (orig. Microsoft Visual Basic step by step). Helsinki: Readme.fi.

20 DoCmd.TransferSpreadsheet Method. Verkkodokumentti. Microsoft. <http://msdn.microsoft.com/en-us/library/office/bb214134(v=office.12).aspx> Luettu 8.4.2013.

21 AcSpreadSheetType Enumeration. Verkkodokumentti. Microsoft. <http://msdn.microsoft.com/en-us/library/office/bb225982(v=office.12).aspx> Luettu 8.4.2013.

TIETOKANNAN TOIMINTOJA OHJAAVAN LOMAKKEEN RAKENNE

```
Private Sub Form_Load()

    ' TUODAAN EXCELEISTÄ TARVITTAVAT LOMAKKEET TAULUIKSI ACCESSIIN !!
    ' HUOMAA, ETTÄ LOMAKKEIDEN NIMIEN (Sheet1 yms.) PITÄÄ OLLA YKSILÖIVIÄ,
    ' KOSKA MUUTEN NE TALLENTUVAT AIKAISEMMIN ACCESSIIN IMPORTOIDUN TAULUN PÄÄLLE..

    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "osaluettelo ja hinnat", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "PalkkaData", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "ADDITIONAL WARRANTY ORDERS", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "Closed complaints", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "Purchase Claims", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "SCRAP REPORT", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "USA WARRANTY PARTS", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "WARRANTY HOURS", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "WARRANTY PARTS", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "Warranty shipping cost", "C:\xxx", True, ""
    DoCmd.TransferSpreadsheet acImport, acSpreadsheetTypeExcel9, "CREDITED WARRANTY", "C:\xxx", True, ""


    ' AJETAAN KYSELYT, JOILLA KASATAAN LOPULLISTA DATA-TAULUA !!
    ' KYSELYT KOHDISTUVAT AIKAISEMMIN IMPORTILLA TUOTUIHIN TAULUIHIN !!
    DoCmd.OpenQuery "Part_No extension fix1", acViewNormal, acEdit
    DoCmd.OpenQuery "Part_No extension fix2", acViewNormal, acEdit
    DoCmd.OpenQuery "Closed complaints Query", acViewNormal, acEdit
    DoCmd.OpenQuery "PalkkaData Query", acViewNormal, acEdit
    DoCmd.OpenQuery "Part No fake extensions", acViewNormal, acEdit
    DoCmd.OpenQuery "Purchase Claims Query", acViewNormal, acEdit
    DoCmd.OpenQuery "SCRAP REPORT Query", acViewNormal, acEdit
    DoCmd.OpenQuery "USA WARRANTY PARTS Query", acViewNormal, acEdit
    DoCmd.OpenQuery "WARRANTY HOURS Query", acViewNormal, acEdit
    DoCmd.OpenQuery "WARRANTY PARTS Query", acViewNormal, acEdit
    DoCmd.OpenQuery "Warranty shipping cost Query", acViewNormal, acEdit
    DoCmd.OpenQuery "CREDITED WARRANTY Query", acViewNormal, acEdit
    DoCmd.OpenQuery "All tables in one 2 query", acViewNormal, acEdit


    ' SIIRRETÄÄN LOPULLINEN DATA-TAULU EXCELIIN
    DoCmd.TransferSpreadsheet acExport, acSpreadsheetTypeExcel12Xml, "All tables in one 2", "T:\xxx", True, ""


    ' AVATAAN KYSEINEN EXCEL-TIEDOSTO, JOSSA LOPULLISEN TAULUN KÄSITTELYÄ
    Program = "Excel.exe T:\Quality\Quality\Poor_quality_metrics\PQC_METER-NEW_VERSION\Pivot_Data\PivotData.xlsm"
    TaskID = Shell(Program, vbNormalFocus)

    ' TYHJENNETÄÄN TAULUKOT DATASTA TILAN SÄÄSTÄMISEKSI
    DoCmd.OpenQuery "Delete ADDITIONAL WARRANTY ORDERS", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete Closed complaints", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete Palkkadata", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete Purchase claims", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete SCRAP REPORT", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete WARRANTY HOURS", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete WARRANTY PARTS", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete USA WARRANTY PARTS", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete Warranty shipping cost", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete CREDITED WARRANTY", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete All tables in one", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete All tables in one 2", acViewNormal, acEdit
    DoCmd.OpenQuery "Delete Part Cost", acViewNormal, acEdit

    ' SULJETAAN TIETOKANTA JA SIIRRYTÄÄN EXCELIIN
    DoCmd.Quit acQuitSaveAll

End Sub
```

TUOTEPERHEEN MÄÄRITTELEVÄ VBA-FUNKTIO

```
Function getProductLine(product As String)
Dim temppi As String
Dim Device As String
temppi = ""

' ETSITÄÄN HAETUISTA RIVISTÄ MÄÄRITELTYJÄ HAKUSANOJA. JOS VASTAAVUUS LÖYTYY, KIRJOITTUU
' KYSEISEN KOHDAN TEMPPI:N DATA TUOTELINJAKSI. ESIM JOS LAITTEEN NIMI SISÄLTÄÄ "OP30" OSAA
' HAKUPROSESSI KIRJOITTAA TUOTELINJAN KOHDALLE "ELEO"

' JOS "HAKUKONEESEEN" HALUTAAN LISÄTÄ HAKUSANOJA ESIM UUSIEN LAITTEIDEN JULKAISUN VUOKSI,
' LISÄÄ HAKUSANA KYSEISEN PERHEEN SISÄÄN "PRODUCT LIKE "*XXX*" OR _ " MALLIN MUKAISESTI.
' HUOM! VARMISTA ETTÄ THEN PÄÄTTYINEN RIVI ON VIIMEINEN HAKUSANAN SISÄLTÄMÄ RIVI!!!

' UUSIA TUOTEPERHEITÄ VOIDAAN MYÖS LISÄTÄ NIIN PALJON KUIN HALUAA KOPIOIMALLA JOKU ALLA
' OLEVISTA HAUISTA JA VAIHTAMALLA HAKUSANAT SEKÄ LINJAN NIMI (TEMPPI)

‘ MEKANIIKAN HAKU
If product Like "*MECHANIC*" Then
temppi = "MECHANICS"
End If

‘ HV/ELEKTRONIIKKALINJAN HAKU
If product Like "*HV*" Or _
product Like "*ELECTR*" Then
temppi = "ELECTRONICS/HV"
End If

‘ VARAOSIEN HAKU
If product Like "*SPARE*" Then
temppi = "SPARE PARTS"
End If

‘ LOPETETTUJEN LAITTEIDEN HAKU
If product Like "*PCT*" Or _
product Like "*TOME*" Then
temppi = "OTHER"
End If

‘ CLASSIC LINJAN HAKU
If product Like "*SCANORA*3D*" Or _
product Like "*CRANEX*" Or _
product Like "*FOCUS*" Or _
product Like "*MINRAY*" Then
temppi = "CLASSIC"
End If

‘ OHJELMISTOJEN HAKU
If product Like "*SOFTWARE*" Or _
product Like "*WINDOWS*" Or _
product Like "*DFW*" Or _
product Like "*CLINIVIEW*" Then
temppi = "SOFTWARE"
End If

‘ SENSORILINJAN HAKU
If product Like "*SNAPSHOT*" Or _
product Like "*SENSOR*" Or _
product Like "*SIGMA*" Or _
product Like "*TOTO*" Then
temppi = "SENSORS"
End If

‘ KUVANLUKIJALINJAN HAKU
If product Like "*OPTIME*" Or _
product Like "*EXPRESS*" Or _
product Like "*OEM*" Or _
product Like "*SOPRO*" Or _
product Like "*SCAN*EXAM*" Or _
```

```
product Like "*GXPS*500*" Or _
product Like "*PSP*" Or _
product Like "*INTRA*CR*" Then
temppi = "DII"
End If

‘ VANHOJEN PANORAAMA JA KEFFILAITELINJAN HAKU
If product Like "*CRANEX*D*" Or _
product Like "*OP*200*" Or _
product Like "*OP*100*" Or _
product Like "*OC*100*" Or _
product Like "*SMART*PAD*" Or _
product Like "*VT*" Or _
product Like "*OC*200*" Then
temppi = "OLD PANS & CEPHS"
End If

‘ ELEO LINJAN HAKU
If product Like "*OP30*" Or _
product Like "*GXDP*300*" Or _
product Like "*PAN*EXAM*" Or _
product Like "*NOVUS*" Then
temppi = "ELEO"
End If

‘ HLEO LINJAN HAKU
If product Like "*OP*300*" Or _
product Like "*OC*300*" Or _
product Like "*700*" Or _
product Like "*PAN*PLUS*" Or _
product Like "*CRANEX*3D*" Or _
product Like "*NGEO*" Then
temppi = "HLEO"
End If

getProductLine = temppi
End Function
```